2

AD-A226 660

NAVAL POSTGRADUATE SCHOOL

Monterey, California

UNITED STATES NAVY
NAVAL POSTGRADUATE SCHOOL

THESIS

REQUIREMENTS ANALYSIS
FOR A LOW COST
COMBAT DIRECTION SYSTEM

James A. Seveney

Guenter P. Steinberg

June 1990

Thesis Advisor: Luqi

UNCLASSIFIED
SECURITY CLASSIFICATION OF THIS PAGE

REPORT DOCUMENTATION PAGE

1a REPORT SECURITY CLASSIFICATION UNCLASSIFIED	1b. RESTRICTIVE MARKINGS
2a SECURITY CLASSIFICATION AUTHORITY	3. DISTRIBUTION/AVAILABILITY OF REPORT Approved for public release; distribution is unlimited
2b DECLASSIFICATION/DOWNGRADING SCHEDULE	
4 PERFORMING ORGANIZATION REPORT NUMBER(S) NPS52-90-023	5. MONITORING ORGANIZATION REPORT NUMBER(S)
6a. NAME OF PERFORMING ORGANIZATION Computer Science Dept. Naval Postgraduate School; 6b. OFFICE SYMBOL (if applicable) 52	7a. NAME OF MONITORING ORGANIZATION Naval Postgraduate School
6c ADDRESS (City, State, and ZIP Code) Monterey, CA 93943	7b. ADDRESS (City, State, and ZIP Code) Monterey, CA 93943
8a. NAME OF FUNDING/SPONSORING ORGANIZATION Naval Postgraduate School; 8b. OFFICE SYMBOL (if applicable)	9. PROCUREMENT INSTRUMENT IDENTIFICATION NUMBER O&MN, Direct funding
8c ADDRESS (City, State, and ZIP Code) Monterey CA 93943	10. SOURCE OF FUNDING NUMBERS: PROGRAM ELEMENT NO; PROJECT NO.; TASK NO.; WORK UNIT ACCESSION NO

11 TITLE (Include Security Classification)
Requirements Analysis for a Low Cost Combat Di..ction System

12 PERSONAL AUTHOR(S)
Seveney, James Arthur; Steinberg, Guenter Peter

13a TYPE OF REPORT	13b. TIME COVERED	14. DATE OF REPORT (Year, Month, Day)	15. PAGE COUNT
Master's Thesis	FROM 05/89 TO 06/90	June 1990	252

16 SUPPLEMENTARY NOTATION

17 COSATI CODES			18. SUBJECT TERMS (Continue on reverse if necessary and identify by block number)
FIELD	GROUP	SUB-GROUP	Combat Direction Systems, Software Engineering, Requirements Analysis, Man-Machine Interface, Tactical Workstations

19 ABSTRACT (Continue on reverse if necessary and identify by block number)

The U.S. Navy has recently embarked on a program called *Next Generation Computer Resources* (NGCR) whose aim is a cooperative effort between Navy and industry to field a set of state-of-the-art computers for shipboard use in the late 1990's. One of the important features of NGCR is the use of commercial hardware developed to a standardized protocol and ruggedized for military applications. This protocol provides for compatibility between machines. The machines themselves may be of any architecture so long as they can meet the protocol requirements as specified by NGCR. The NGCR protocol, while not fully defined, implies the use of microprocessor based workstations.

The Naval Sea Systems Command, in an effort to keep pace with combat system software, desires experience in developing software targeted for commercially available NGCR machines (workstations). This study provides a detailed set of initial requirements for such a system. The approach is to implement the basic features of a Combat Direction System (CDS) on a microprocessor based workstation. This *low cost* CDS (LCCDS) will initially be installed on non-combatant vessels which currently have no computer processing capability at all. Eventually the LCCDS may be used to augment current processing capability of CDS equipped combatants followed by future systems designed around high speed (SAFENET) networks of *tactical workstations.*

20. DISTRIBUTION/AVAILABILITY OF ABSTRACT ☒ UNCLASSIFIED/UNLIMITED ☐ SAME AS RPT. ☐ DTIC USERS	21. ABSTRACT SECURITY CLASSIFICATION UNCLASSIFIED
22a NAME OF RESPONSIBLE INDIVIDUAL Luqi	22b. TELEPHONE (Include Area Code) (408) 646-2735; 22c. OFFICE SYMBOL 52Lq

DD FORM 1473, 84 MAR — 83 APR edition may be used until exhausted. All other editions are obsolete

SECURITY CLASSIFICATION OF THIS PAGE
UNCLASSIFIED

REQUIREMENTS ANALYSIS FOR A LOW COST COMBAT DIRECTION SYSTEM

by
James Arthur Seveney
Lieutenant Commander, United States Navy
and
Guenter Peter Steinberg
Lieutenant Commander, Federal German Navy
Dipl.-Ing.(Fh), Hochschule der Bundeswehr Muenchen, 1980

Submitted in partial fulfillment of the
requirements for the degree of

MASTER OF SCIENCE IN COMPUTER SCIENCE

from the

NAVAL POSTGRADUATE SCHOOL
June 1990

Authors: [illegible] [illegible]
James A. Seveney   Guenter P. Steinberg

Approved By: [illegible]
Luqi, Thesis Advisor

[illegible]
Valdis Berzins, Second Reader

[illegible]
Robert B. McGhee, Chairman,
Department of Computer Science

ABSTRACT

The U. S. Navy has recently embarked on a program called *Next Generation Computer Resources* (NGCR) whose aim is a cooperative effort between Navy and industry to field a set of state-of-the-art computers for shipboard use in the late 1990's. One of the important features of NGCR is the use of commercial hardware developed to a standardized protocol and ruggedized for military applications. This protocol provides for compatibility between machines. The machines themselves may be of any architecture so long as they can meet the protocol requirements as specified by NGCR. The NGCR protocol, while not fully defined, implies the use of microprocessor based workstations.

The Naval Sea Systems Command, in an effort to keep pace with combat system software, desires experience in developing software targeted for commercially available NGCR machines (workstations). This study provides a detailed set of initial requirements for such a system. The approach is to implement the basic features of a Combat Direction System (CDS) on a microprocessor based workstation. This *low cost* CDS (LCCDS) will initially be installed on non-combatant vessels which currently have no computer processing capability at all. Eventually the LCCDS may be used to augment current processing capability of CDS equipped combatants followed by future systems designed around high speed (SAFENET) networks of *tactical workstations*.

THESIS DISCLAIMER

This thesis uses a number of names which are trademarks of various corporations. In this section we give the appropriate credits.

- Ada is a trademark of the United States Government Ada Joint Program Office.
- GemStone is a trademark of Servio Logic Corporation.
- SOLARIS is a trademark of Genesco Corp.
- StP is a trademark of Integrated Design Environments, Inc.
- SUN Workstation is a trademark of Sun Microsystems.
- UNIX is a trademark of AT & T.
- X Windows is a trademark of MIT.

The views expressed in this thesis are those of the authors and do not reflect the official policy or position of the Department of Defense or the U. S. Government.

TABLE OF CONTENTS

I. INTRODUCTION 1
A. HISTORICAL BACKGROUND: NTDS 1
B. COMBAT SYSTEMS 3
1. Combat Direction Systems (CDS) 3
2. Low Cost Combat Direction System (LCCDS) 4
C. NAVY COMMAND AND CONTROL SYSTEMS 6
D. FORMAL SOFTWARE ENGINEERING APPROACH 8
E. RESEARCH OBJECTIVES 9
F. ORGANIZATION 10

II. LCCDS REQUIREMENTS ANALYSIS 11
A. LCCDS TASK OVERVIEW 11
1. Features of a Combat Direction System 11
2. LCCDS Definition 12
B. INITIAL PROBLEM STATEMENT 15
1. Increment One 15
2. Increment Two 17
3. Increment Three 18
4. Increment Four 19
5. Increment Five 19
C. ENVIRONMENT MODEL 20
1. Preface 20
2. Terms and Definitions 22
D. LCCDSWS GOALS 25
E. LCCDSWS PROTOTYPE CONSTRAINTS 28
1. Resource Constraints 29
2. Implementation Constraints 29
3. Performance Constraints 32
4. Tactical Data Representation Constraints 33
5. MMI Compatibility Constraints 34
F. LCCDSWS GOAL/FUNCTION REFINEMENT 35
1. Man-Machine Interface (MMI) Function 35
2. System Control Function 74
3. The Tactical Database Function 77
4. The Basic CDS Functions 89

III. LCCDS ESSENTIAL MODEL 105
A. THE ENVIRONMENTAL MODEL 106
1. The Statement of Purpose 106
2. The Context Diagram 106
3. The Event List 107

B. THE BEHAVIORAL MODEL 109
1. Data Flow Diagrams 109
2. Data Structure Diagrams 113
2. Data Dictionary 133

IV. MAN-MACHINE INTERFACE DESIGN GUIDELINES 179
A. LCCDS USER CONCEPTUAL MODEL 179
B. FUNCTIONS OF A TACTICAL MAN-MACHINE INTERFACE 182
C. COLOR USE AND SCHEMES 186
D. AURAL CUES AND ALERTS 189
E. SPEECH RECOGNITION 191
1. Speaker Dependent 192
2. Speaker Independent 193
3. Discrete Speech 193
4. Connected Speech 194
5. Applications for Speech Recognition 194
6. Conclusions on Speech Recognition 194
F. AN EXTENSIBLE MMI 195

V. CONCLUSIONS 198
A. RECOMMENDATIONS 198
1. Formal Approach 198
2. Rapid Prototyping 199
3. Further Research 201
B. EVOLUTION OF THE LCCDS 203
1. Modifications to the Basic LCCDSWS 203
2. Extensions of the LCCDS Functionality 207
3. LCCDS Future Versions 210
C. LCCDS FINAL CONSIDERATIONS 215

APPENDIX A (STATEMENT OF WORK FOR LCCDS) 218

APPENDIX B (TABLE OF REFINED GOALS) 223

LIST OF REFERENCES 230

INITIAL DISTRIBUTION LIST 233

LIST OF FIGURES

Figure 1: Navy Command and Control Structure 7
Figure 2: Software Development Activity Cycle 8
Figure 3: Functional Dependency for CDS Software Functions 12
Figure 4: Major Components of a Ship's Combat System 13
Figure 5: LCCDS Components 14
Figure 6: Context Diagram for LCCDSWS, Increments I & II 17
Figure 7: Context Diagram for LCCDSWS, Increments I, II & III 18
Figure 8: Context Diagram for LCCDSWS, Increments I, II, III & IV 19
Figure 9: Context Diagram for LCCDSWS, Increments I through V 20
Figure 10: Initial Context Diagram for LCCDS 21
Figure 11: Man-Machine Interface Structure 37
Figure 12: Basic Interactive Window Format 42
Figure 13: Standard Support Window with Check Boxes, Radio and Push Buttons 44
Figure 14: Graphics Application and Support Windows 46
Figure 15: Document Window with Scroll Bars and Warning Window 47
Figure 16: LCCDS Root Window 50
Figure 17: The Tactical Plot Window 51
Figure 18: Radar and Video Select Window 52
Figure 19: Example Function I/O Window 52
Figure 20: Standard Document (New Track Establishment) Window 53
Figure 21: LCCDS Track Symbology 56
Figure 22: Velocity Leader 58
Figure 23: The Tactical Plot Range Scale 61
Figure 24: Display Doctrine Syntactic Structure 68

Figure 25: Boolean Expression Structure 70
Figure 26: Display Doctrine Statement Construction 73
Figure 27: Index Tableau Support Window 100
Figure 28: The Context Diagram (from Figure 10, page 21) 106
Figure 29: The Top Level Diagram 111
Figure 30: The Zero Level Diagram 112
Figure 31: Data Structure Diagram - Additional_TrackInfo 113
Figure 32: Data Structure Diagram - CPA_Data 114
Figure 33: Data Structure Diagram - CPA_Request 114
Figure 34: Data Structure Diagram - Display_Tracks 115
Figure 35: Data Structure Diagram - Geographic_Position 115
Figure 36: Data Structure Diagram - Latitude 116
Figure 37: Data Structure Diagram - Link 116
Figure 38: Data Structure Diagram - Link11_Enable 117
Figure 39: Data Structure Diagram - Link11_Tracks 117
Figure 40: Data Structure Diagram - LinkTrackInfo 118
Figure 41: Data Structure Diagram - Load_Map 118
Figure 42: Data Structure Diagram - Longitude 119
Figure 43: Data Structure Diagram - Manual_Tracks 119
Figure 44: Data Structure Diagram - Map_Data 120
Figure 45: Data Structure Diagram - Map_Messages 120
Figure 46: Data Structure Diagram - Map_Request 121
Figure 47: Data Structure Diagram - NavStatus 121
Figure 48: Data Structure Diagram - Nav_Enable 122
Figure 49: Data Structure Diagram - Navigation_Data 122
Figure 50: Data Structure Diagram - Navigational_Messages 123

Figure 51: Data Structure Diagram - Operator_Commands 123

Figure 52: Data Structure Diagram - Operator_Messages 124

Figure 53: Data Structure Diagram - Ownship 124

Figure 54: Data Structure Diagram - Radar_Enable 125

Figure 55: Data Structure Diagram - Radar_Info 125

Figure 56: Data Structure Diagram - Radar_Tracks 126

Figure 57: Data Structure Diagram - Range_Select 126

Figure 58: Data Structure Diagram - Special_Points 127

Figure 59: Data Structure Diagram - System_InitData 127

Figure 60: Data Structure Diagram - System_Messages 128

Figure 61: Data Structure Diagram - System_Tracks 128

Figure 62: Data Structure Diagram - TacInfo_Alpha 129

Figure 63: Data Structure Diagram - TacInfo_Graphic 129

Figure 64: Data Structure Diagram - Time 130

Figure 65: Data Structure Diagram - Track_Data 130

Figure 66: Data Structure Diagram - Track_Status 131

Figure 67: Data Structure Diagram - Waypoint_Request 131

Figure 68: Data Structure Diagram - Waypoint_Data 132

Figure 69: Data Structure Diagram - Vehicular_Tracks 132

Figure 70: Simplified Model of Information Conversion 185

Figure 71: Non-Combatant LCCDS 210

Figure 72: Non-Combatant LCCDS with CIWS 211

Figure 73: Small-Combatant LCCDS 212

Figure 74: LCCDS as CDS Tactical Workstation 213

Figure 75: Future CDS - Distributed Architecture using Tactical Workstations 214

TABLE OF ABBREVIATIONS

ACS	Afloat Correlation System
ASMD	Anti-Ship Missile Defense
BF/BG	Battle Force/Battle Group
CAPS	Computer Aided Prototyping System
CDSWS	CDS Software System
CIWS	Close-In Weapon System
CNO	Chief of Naval Operations
CPA	Closest Point of Approach
CS	Computer System
DLRP	Data Link Reference Point
DMA	Defense Mapping Agency
DoD	Department of Defense
DTED	Digital Terrain Elevation Data
EMCON	Emission Control
ESM	Electronic Support Measures
ETA	Estimated Time of Arrival
ETE	Estimated Time Enroute
EW	Electronic Warfare
EWC	Electronic Warfare Coordinator
FCDSSA	Fleet Combat Direction System Support Activity
FCS	Fire Control System
GC	Graphic Context
GFE	Government Furnished Equipment
HASC	House Armed Services Committee
HWS	Harpoon Weapon System
IDS	Interface Design Document
I/O	Input/Output
JOTS	Joint Operational Tactical System
LCCDS	Low Cost Combat Direction System
LCCDSWS	Low Cost Combat Direction System Software
MMI	Man-Machine Interface

NAVAIR	Naval Air Systems Command
NAVSEA	Naval Sea Systems Command
NCCS-A	Navy Command and Control Station - Afloat
NGCR	Next Generation Computer Resources
NM	Nautical Miles
NPS	Naval Postgraduate School
NRAC	Naval Research Advisory Committee
NSSMS	NATO Sea Sparrow Missile System
NTDS	Navy Tactical Data System
OODBMS	Object-Oriented Database Management System
PIM	Position and Intended Movement
POST	Prototype Ocean Surveillance System
PPI	Plan Position Indicator
PSDL	Prototype System Design Language
PU	Link 11 Participating Unit
ROC	Required Operational Capability
SATNAV	Satellite Navigation
SNDC	Standard Navy Display Console
SOP	Standard Operating Procedures
SPAWAR	Space and Naval Warfare Systems Command
TAO	Tactical Action Officer
TBD	To Be Determined
TFCC	Tactical Flag Command Center
TN	Link 11 Track Number
WVS	World Vector System

ACKNOWLEDGEMENTS

We owe a great deal to many people who have given us help, ideas and encouragement during the process of writing this thesis. Special thanks go to our families, for their support and patience. Also, we would like to mention a few folks individually for the time, energy or advise they lent on behalf of our efforts to formulate our concepts, research them and finally get it all down on paper:

CPT Pat Barnes, USAF (NPS, Monterey)

Mr. Bob Duffy, (NAVSEA 61T, Washington, DC)

CDR Wayne Duke, USN (NOSC, San Diego)

CDR Rich Gragg, USN (OP-942F4, Washington DC)

Mr. William Hanley (FCDSSA, San Diego)

LCDR John Hearing, USN (NPS, Monterey)

Dr. Chuck Hutchins (NPS, Monterey)

LCDR Scott Kelly, USN (NAVSEA 61T, Washington, DC)

MAJ Mike Nelson, USAF (NPS, Monterey)

Dr. Glenn Osga (NOSC, San Diego)

Dr. Gary Poock (NPS, Monterey)

Mr. George Roberson (FCDSSA, San Diego)

LCDR Jeff Schweiger, USN (NPS, Monterey)

LCDR John Yurchak, USN (NPS, Monterey)

I. INTRODUCTION

The Low Cost Combat Direction System (LCCDS) project, sponsored by the Naval Sea Systems Command (NAVSEA), will implement the basic features of a Combat Direction System on a commercially available microprocessor-based workstation. One of the intended uses for the system is on-board non-combatant ships where no automated combat information processing capability currently exists.

The increasing availability of small, powerful and relatively inexpensive microprocessor workstations, ruggedized for military applications, makes some form of Combat Direction System (CDS) feasible for any Navy vessel. The U. S. Navy Combat Logistics Force (CLF) ships and other fleet support vessels currently handle all tactical navigation, contact management and intelligence tasks manually using only maneuvering board and grease pencil. The LCCDS will be capable of filling this information processing void, at a fraction of the cost of a full tactical CDS.

The prime objective of this thesis is to define the environment, goals, functions and constraints of a prototype LCCDS within the context of current CDS system requirements. This is accomplished through a formal requirements analysis process, using established software engineering methodology, to derive the proposed LCCDS software system as a small subset of existing Combat Direction Systems.

A. HISTORICAL BACKGROUND: NTDS

Since World War II, the development of radar has produced spectacular changes in the conduct of naval warfare. Jet aircraft replaced their propeller driven predecessors, and the rate of combat information increased exponentially. The magnitude of information that needed to be assembled overwhelmed conventional "grease pencil" means and there was an obvious need for some type of system or technique to gather operational data, process the data into meaningful and useful information, and present the information in formats that facilitated decision making by those to whom it was presented.

In the late 1950's, with the inception of the Navy Tactical Data System (NTDS), the U. S. Navy began using computers in tactical shipboard systems. The Naval Tactical Data System evolved from this need to normalize and convert increasing amounts of information from multiple sources (e.g., sensors, navigation data, observation) to common representations that could be processed, stored, and disseminated to shipboard tactical users so that weapons employment decisions could be made more effective with less reaction time.

When NTDS was first deployed, and during the decade of the 60's its role was tactical data integration. Tactical information generated was inconsistent, mirroring the inefficiencies and inaccuracies of an automated design concept largely manually implemented. NTDS was forced to interface with existing subsystem designs that were subject to little more than minor accommodations to it. The NTDS design, as a result, represented a compromise heavily biased toward data conversion and limited to serving as a conduit through which personnel collected, processed and acted on tactical information. Uncoordinated changes in the interfacing subsystems caused a continual catch-up mode. [Ref. 1]

Further study and the experience of more than two decades of NTDS history resulted in the development of Combat Direction Systems with a substantial increase in capabilities over the NTDS.

B. COMBAT SYSTEMS

A ship's combat system is typically thought of as men and machines which fight the ship. This could be seen as both weapons and the means by which weapons are controlled. Subsystems such as navigation and communications, which are not directly involved in weapons control, also play an important part in current combat systems. The combat system can then be seen as the payload carried by a Navy warship. It is the composite of shipboard elements and personnel that includes, either actively or passively, detection, tracking, identification, processing, evaluation and control of engagement of hostile threats. In terms of equipment, the combat system may include subsystems containing missiles, guns, sensors, electronic warfare (EW), navigation and communication systems.

1. Combat Direction Systems (CDS)

A Ship's Combat Direction System is organized to direct combat. Its resources include personnel, equipment (hardware and software), communications and logistic support facilities. The CDS and the sensor and weapons systems it controls are part of the Combat System. CDSs are defined as follows [Ref. 2:p. 3-5]:

> Those combinations of data and men handling the systems, either manual or automated, employed to execute the combat direction functions. They support command at levels from the platform (ships, aircraft, submarines) up to, and including, task group/force.
>
> A CDS consists of a complex set of data inputs, user consoles, converters, adapters and radio terminals interconnected with high speed general purpose computers and their stored programs. Combat data are collected, processed and composed into a picture of the overall tactical situation that enables the force commander to make rapid and accurate decisions.

A high level goal for the CDS can be stated as follows: Optimize the individual ship's fighting capability in concert with the other segments of the ship's combat system [Ref. 2:p. 3-7].

In general, the CDS role is composed of the following [Ref. 2:p.3-7]:

(1) The automated shipboard database manager of tactically significant tracks.

(2) The primary combat system element supporting the Combat Direction Center (CDC).

2. Low Cost Combat Direction System (LCCDS)

Recently the U. S. Navy has initiated the Next Generation Computer Resource (NGCR) program which is a cooperative effort between Navy (NRAC) and industry to field a set of state-of-the-art computers for shipboard use in the late 1990's [Ref.3]. In their briefings with CNO, SPAWAR, NAVAIR, NAVSEA, HASC and numerous DoD contractors, the NGCR Panel made the following recommendations:

- The Navy should mandate widely used commercial standards for its computing resources. The Navy should resist the temptation to have its own unique standards.

- The Navy should encourage the use of ruggedized equipment. Many mission critical systems operate in protected environments where full militarization is too much of a price to pay for up-to-date performance.

- The Navy should move toward rapid elimination of Government Furnished Equipment (GFE) status of the UYK computers.... Rewriting existing UYK software in Ada should be considered seriously.

- The Navy should mandate standards at the system level only. Mandating Navy-wide standards at a lower level can be counterproductive.

- The Navy should reorient its planned prototyping effort. The purpose should be to demonstrate commercial standards at work and to support the upgrading of computing capability on current ships.

In particular, the prototyping of a weapon system using ruggedized equipment and commercial standards was recommended. In preparation for NGCR availability, NAVSEA wishes to gain some experience in the development of CDS software on commercially available machines [Ref. 4]:

> The Navy desires to develop a Low Cost Combat Direction System (LCCDS) that can potentially be installed on non-combatant ships or to augment the processing capability on board current CDS equipped ships. An implementation in Ada is desired. A total of five increments will result in a continuously improving product that reflects the desired features of the CDS community. [Ref. 4]

The specific features of the five increments are as follow:

1. Select architecture and software support environment for the LCCDS software system. Integrate object oriented database support and graphics capability.

2. Integrate manual tracking and identification capability.

3. Integrate a receive-only Link 11 capability.

4. Integrate an own ship maneuvering capability.

5. Integrate an automatic tracking capability.

Enclosure 1 to Reference 4, *Statement of Work for the Low Cost Combat Direction System (LCCDS)*, provides an informal, high level description and is included as Appendix A.

C. NAVY COMMAND AND CONTROL SYSTEMS

The navy has committed much effort to the design and development of a standard definition of the command and control task. Specifically, the Naval Space and Warfare Systems Command (SPAWAR) has been developing the Navy Command and Control System Afloat (NCCS-A) program. The NCCS-A program encompasses several command and control systems including the Tactical Flag Command Center (TFCC), Joint Operational Tactical System (JOTS), Afloat Correlation System (ACS) and the Prototype Ocean Surveillance System (POST). The purpose of this program is to provide the Battle Force/Battle Group (BF/BG) Commander and his subordinate warfare commanders/coordinators a consistent, *near real-time*, tactical picture from all-source data, the ability to access and manipulate the related tactical data, and the decision support tools required to effectively execute coordinated Command and Control [Ref. 5:p. 1].

CDS and NCCS-A differ greatly in the areas of performance constraints and data generation and manipulation. CDS's are deadline driven systems requiring *hard real-time* performance. Different functionality is necessary in a CDS because it must support a lower level of tactical decision-making needs and provide physical control for external systems.

The scope, or perspective, in which the tactical data will be used is different as well. NCCS-A covers a much larger scale, involving both organic and inorganic sources of information. However, *the types of tactical data necessary to each are identical.* See Figure One for an overview of how the CDS fits under the "umbrella" of the larger scale naval command and control systems. The CDS can be viewed as a subset of NCCS-A, dealing with the same types of data, but on a smaller scale involving individual ships relying mainly on their own (organic) sensors.

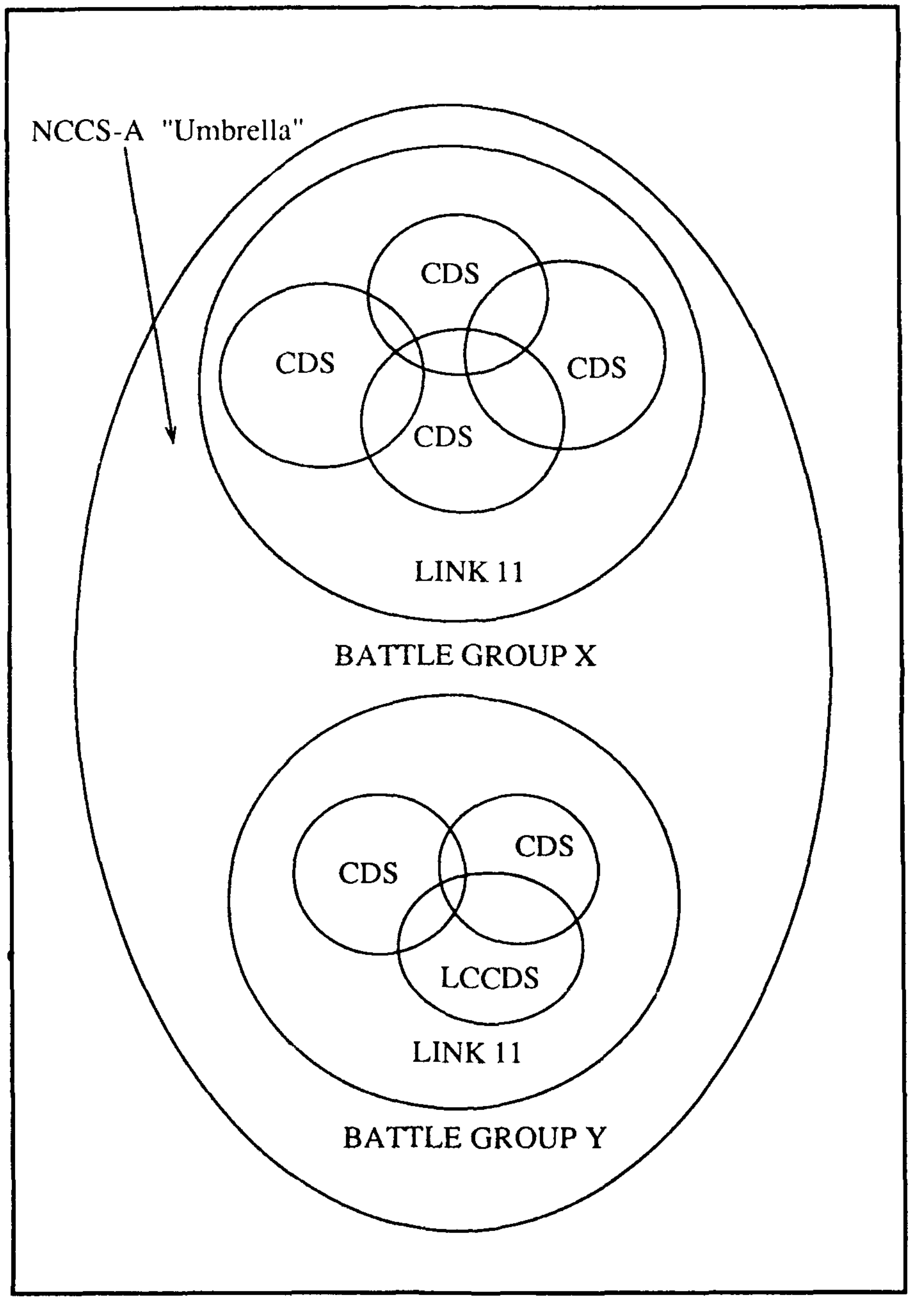


Figure 1: Navy Command and Control Structure

D. FORMAL SOFTWARE ENGINEERING APPROACH

It is widely acknowledged that while the software development process is not a rigid or narrowly defined thing, there are several essential ingredients which form the basis of any serious software development task. As pointed out by Berzins [Ref. 6] these development activities or ingredients blend together to form a cycle. Figure 1 depicts this cycle as a data flow diagram where these activities are pipelined and concurrent. In particular, analysis continues throughout development and evaluation.

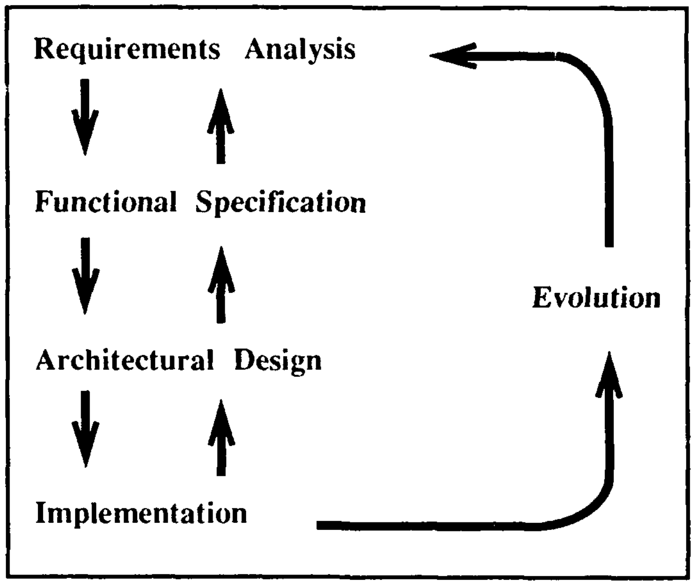


Figure 2: Software Development Activity Cycle

The boundaries between software development activities are fuzzy and not sharply defined. The cycle is evolutionary in nature and never really ends while the system still exists. However, there is a distinct, very clearly established starting place for the software development cycle: *Requirements Analysis*. The goals of requirements analysis are to define the purpose of the proposed software system and to determine the constraints on its development [Ref. 6:p. 1.5]. The requirements analysis for the LCCDS project is based on two assumptions. First, a small, simplified set of requirements can be drawn from existing documentation on currently implemented (combatant) combat direction systems [Ref. 7] where applicable to the non-combatant vessel. Secondly, a large group of knowledgeable, experienced U. S. Navy Surface Warfare Officers are available here at the Naval Postgraduate School who are more than willing to provide the necessary information and advice needed for a detailed, realistic definition of the prospective LCCDS "customer" needs.

E. RESEARCH OBJECTIVES

The LCCDS program is in its infancy. With the exception of the NAVSEA Statement of Work, there exists no documentation on the desired system. The Navy organization primarily concerned with the development and maintenance of CDS software is the Fleet Combat Direction Support Activity (FCDSSA) located in San Diego, CA and Dam Neck, VA. FCDSSA has provided much technical data on currently implemented systems, but has no immediate plans for involvement with LCCDS [Ref. 8]. The obvious first step, therefore, is to provide a detailed definition of LCCDS.

The final product of this initial phase of research is an analysis of LCCDS requirements which accurately reflects the real CDS needs of non-combatant vessels while remaining within the constraints of a prototype research effort. The objective is to provide the following set of results: [Ref. 6:p. 2-1]

(1) A simplified model of the prototype system's environment.

(2) A description of the goals of the system and the functions it must perform.

(3) Performance constraints on the prototype system.

(4) Constraints on the environment and implementation of the prototype system.

(5) Recommended design approaches based on available technology and fleet experience with existing systems.

It is hoped that the analysis provided by this research will be used as the foundation for a prototype development effort for LCCDS based on software engineering principles.

F. ORGANIZATION

Chapter II provides the central element of this research, an analysis of initial requirements in the form of a hierarchical set of high-level and refined system goals for the prototype LCCDS. *The primary emphasis is on a definition of the software system.*

Chapter III provides a high-level structural analysis of the LCCDS using the Yourdon model. It contains the main data flow diagrams and the data dictionary to model the proposed software system.

Chapter IV focuses on the underlying design concepts and issues related to development of an appropriate man-machine interface (MMI).

Chapter V provides further discussion on the formal software engineering approach to large software system development and recommendations for further LCCDS work.

LCDR Seveney focused on requirements analysis for the man-machine interface and overall system design constraints. LCDR Steinberg concentrated on a developing a set of basic CDS requirements and an object definitions for the tactical database. Both contributed to the system architectural analysis in Chapter III and conclusions on the future of the LCCDS in Chapter V.

II. LCCDS REQUIREMENTS ANALYSIS

A. LCCDS TASK OVERVIEW

The system development will follow the steps as outlined by Berzins [Ref. 6]. It is assumed that the reader is familiar with the sequence and purpose of each step. The complexity and size of the system make it necessary to concentrate on the Requirements Analysis as the initial step. A high level problem statement is given in the NAVSEA Statement of Work [Ref. 4:p. 3]:

> The task of the Low Cost Combat Direction System (LCCDS) is to implement the basic features of a Combat Direction System on a commercially available microprocessor based workstation.

This high-level, informal statement depends at least on two concepts from the application area which are not yet very well understood:

- What are the basic features of a Combat Direction System?
- What is the task of the Low Cost Combat Direction System in terms of a Combat Direction System?

At this point it is necessary to provide informal statements on these concepts to refine the initial problem statement and to provide a motivation for a better understanding of the application area.

1. Features of a Combat Direction System

An example of an informal specification for a Combat Direction System is given in References 7 and 9. An individual ship's mission is therein described by Required Operational Capabilities (ROCs) [Ref. 7:p. 3-37]. ROCs are translated into a set functions of the Combat System together with tasks for each function [Ref. 7:p. 3-5].

A subset of the Combat Systems functions is allocated to the CDS, which consists of hardware, software and personnel [Ref. 7:p. 3-4]. A subset of the CDS functions must be performed by the CDS software system (CDSWS). This subset forms the basis of the initial problem statement (Figure 3).

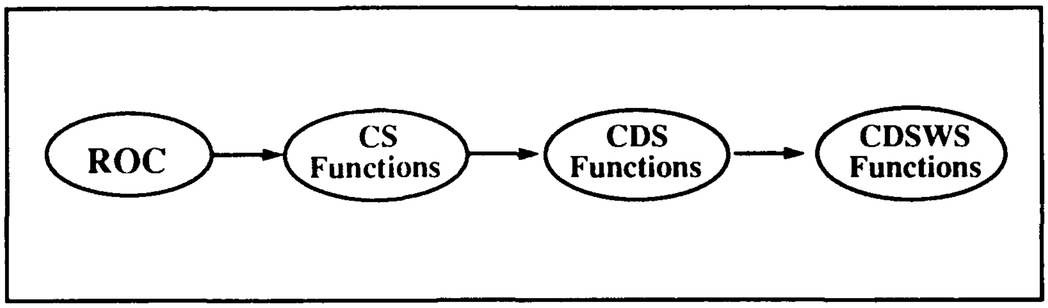


Figure 3 : Functional Dependency for CDS Software Functions

2. LCCDS Definition

Combat Logistics Force (CLF) ships operate as logistic support components of Battle Groups and Battle Forces. Figure 4 shows that Combat Direction Systems (CDS) are the connecting part between the Combat System of ships, Battle Groups and Battle Forces [Ref. 2]. Hence, the Low Cost Combat Direction System has to perform a similar role for CLF ships. The role of Combat Direction Systems in Battle Force Combat Control is defined in Reference 2. A similar definition for the LCCDS is:

> The LCCDS shall be the integrating part between ownship and other units. It shall consist of hardware, software and personnel to provide the ship's command personnel with a facility for monitoring the overall air, surface and subsurface environment. Force and ownship sensor data shall be collected, correlated and evaluated by the LCCDS.

The LCCDS will comprise the same set of components as found in combatant CDS's. The major components of the LCCDS are depicted in Figure 5.

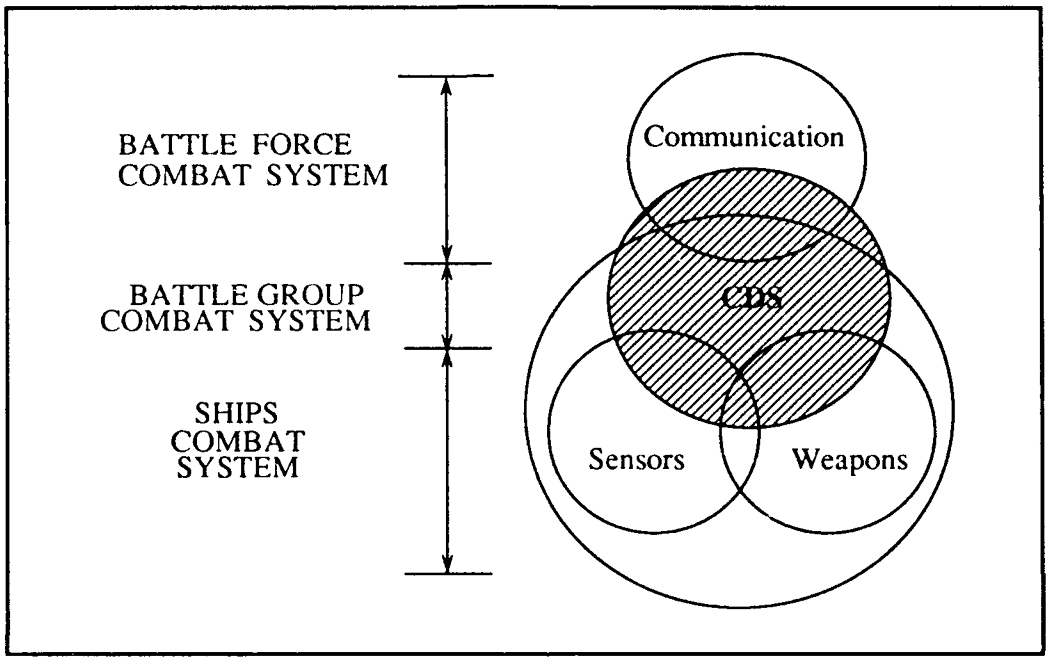


Figure 4 : Major Components of a Ships Combat System

a. LCCDS Hardware

The LCCDS hardware shall consist of a commercially available microprocessor based, ruggidized workstation capable of providing the required data processing and display capability. The workstation must satisfy the system architecture requirements as described in Reference 4. This includes in particular the ability to interface to external systems via bus or network. The data processing facility shall also contain the peripheral equipment necessary to provide the means to load the LCCDS software (e.g., tape, disk, cartridge, or optical disk). The data display facility is comprised of a data display unit and the peripheral equipment to control the LCCDS initialization, configuration and operation of the LCCDS (e.g., keyboard, mouse, trackball, voice input, printer).

A high resolution, bit-mapped 19-inch color monitor shall be used for data display. The data display facility will serve as the principal man-machine interface (MMI) necessary for user command and control of the LCCDS.

b. LCCDS Software

The LCCDS software shall be comprised of an LCCDS Operational Program, Test Programs as necessary and Training Programs. For further details see Reference 7.

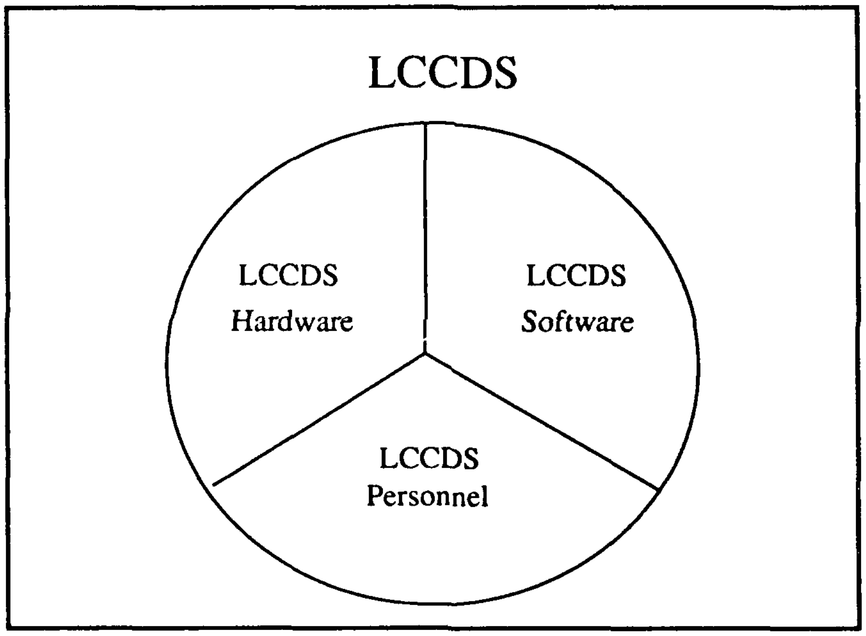


Figure 5 : LCCDS Components

c. LCCDS Personnel

LCCDS personnel are those users assigned to the LCCDS operating station as specified in the ship's Battle Management Organization. For further details see Reference 7, page 3-10.

B. INITIAL PROBLEM STATEMENT

This thesis is intended to focus on the software functionality necessary to implement CDS features on high resolution, bit-mapped workstation display systems. The thrust of this research is to provide a detailed requirements analysis for the software portion of the LCCDS. We refer to this as the ***Low Cost Combat Direction Software System (LCCDSWS)***. The detailed Initial Problem Statement can be defined in terms of a high-level LCCDS program description:

> Develop the prototype of a software system for a Low Cost Combat Direction System (LCCDSWS) that implements the basic features of a Combat Direction System on a commercially available microprocessor based workstation.

The project development shall be performed in five incremental steps as outlined by the NAVSEA *Statement of Work* [Appendix A]. These five steps, in broad high-level terms, are extracted directly from this Statement of Work and reflect NAVSEA's initial development plan and design goals. Figures 6, 7, 8 and 9 depict the LCCDSWS context throughout the five incremental stages of development. The following five sections are extracted from the statement of work and are included to provide a high level perspective of the "customer's" objectives.

1. Increment One

> Select the computer system, run-time operating system and software support environment for the LCCDS. Design/develop or select an existing object oriented database manager that interacts with the entire CDS data base (as it gets defined) and allow the user to define new objects that are not part of the CDS data base. The data base manager should provide features for data entry, user friendly query language for building logical and arithmetic relationships between data base elements, and a powerful output generator for developing display screens and hardcopy formats. Design/develop a display graphics capability which interacts with the

database manager and provides the user with the building blocks necessary to define his own customized screen formats for interactive operation of the system. Display features should include but not be limited to the following:

a. Windows and pull down menus for operation of all program features including the operational features such as mode selection, track information, help commands, display doctrine activation and deactivation.

b. Window and pull down menu selection controlled via mouse, trackball, light pen, touch screen or any other feature deemed usable by the developer.

c. Graphics capability to display all symbology defined in [Appendix A].

d. Ability to overlay the track and ownship position portion of the display with world vector shoreline maps as available from the Defense Mapping Agency (DMA).

All pull down menu options and window spaces should be directly coupled with the data base manager such that all information can be user defined as required. That is, the display capability should be built in generic fashion such that the user can tailor all display presentations including all pull down menu options and window spaces effectively building a new run-time Man Machine Interface (MMI) as desired.

Display doctrine is a feature which enables the user to define the conditions under which data will be displayed and how to present the displayed data. Doctrine statements should be IF - THEN - ELSE type statements where the IF clause provides for operations on tactical information such as type, speed and bearing of tracks to be displayed. The THEN clause provides for the data presentation such as blink the symbol, enlarge, bold type, etc. Doctrine should be simple to define and easily activated or deactivated. Doctrine statements should be allowed to be defined individually and combined, if desired, into a doctrine tree which consists of up to twelve different statements. The IF clause of a doctrine statement should allow for at least twelve qualifiers.

The general response time to any user selected display option should be no greater than one half second.

2. Increment Two

Integrate a Manual Tracking and Identification capability to the basic display capability. Manual Tracking and Identification refers to the ability of the system to user to build and display a set of geographically stable and/or moving points of information of the position portion of the display. Manual Tracking and Identification will include but not be limited to the following features:

a. Maintain the ownship symbol in the center of the position display at all times.

b. Introduce a standard display symbol (see Appendix A) at the current location of the cursor. All symbols should be maintained relative to the ownship symbol.

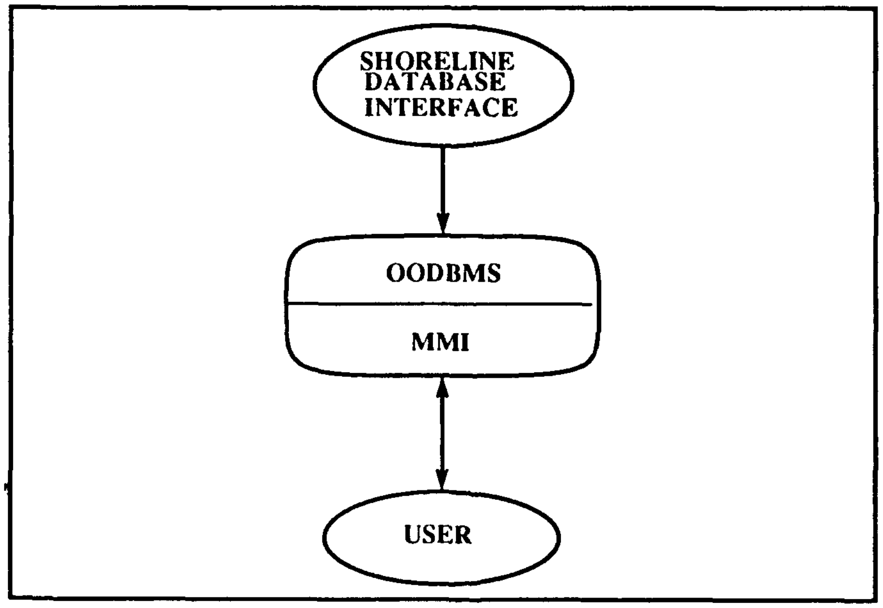


Figure 6 : Context Diagram for LCCDSWS, Increment I and II

c. Allow the user to assign speed and bearing to a vehicular track. Display a proportioned speed leader on the track and change its position at least once every four seconds. Track position should be dead reckoned using the current track bearing.

d. Allow the user to assign additional information to any type of track.

e. Allow a currently displayed track to be hooked (selected) and display selected additional information in a window (e.g.,for a hooked air track display speed, bearing, type, route, etc.). This additional information will be an object from a user defined window.

f. Allow the user to change the Identity of a track (e.g.,from unknown to friendly).

g. Provide for an expandable track file with no artificial size limitations.

3. Increment Three

Include an ownship maneuvering capability which includes navigation capability and track interface information. Ownship maneuvering will include but not be limited to:

a. Allow for the specification of up to six steaming routes with up to 50 waypoints (destinations) per route.

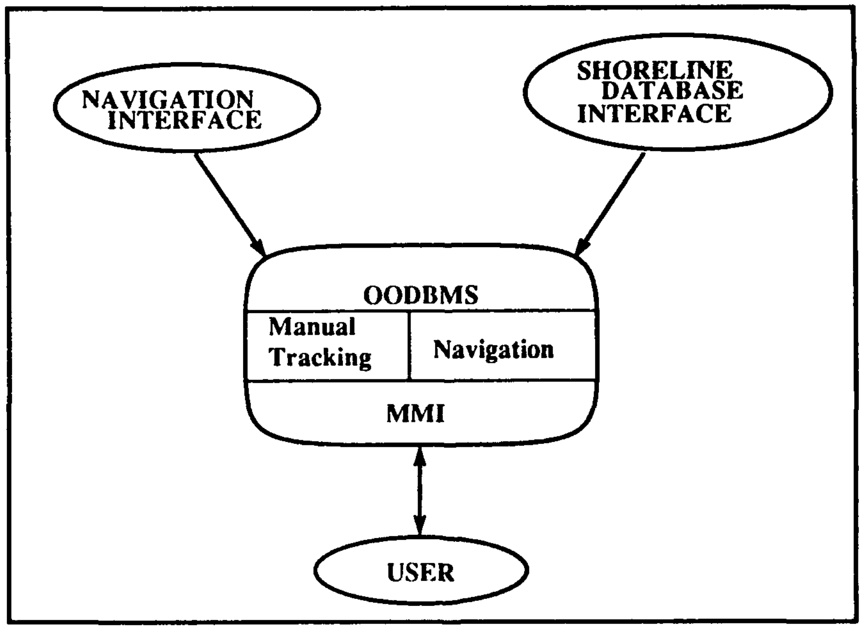


Figure 7 : Context Diagram for LCCDSWS, Increment I, II and III

b. Provide Closest Point of Approach (CPA) geometry from ownship to any track and between any two tracks. Display bearing lines on the position display

4. Increment Four

Integrate an organic auto tracking capability using the (TBD) radar interface. The task entails the development of an interface to a ship mounted radar, parsing the input information and displaying the ship contact information on the position display. This feature should not be costed at this time but the implementation should be considered for easy add on at a later date. A radar IDS will be provided under a different cover.

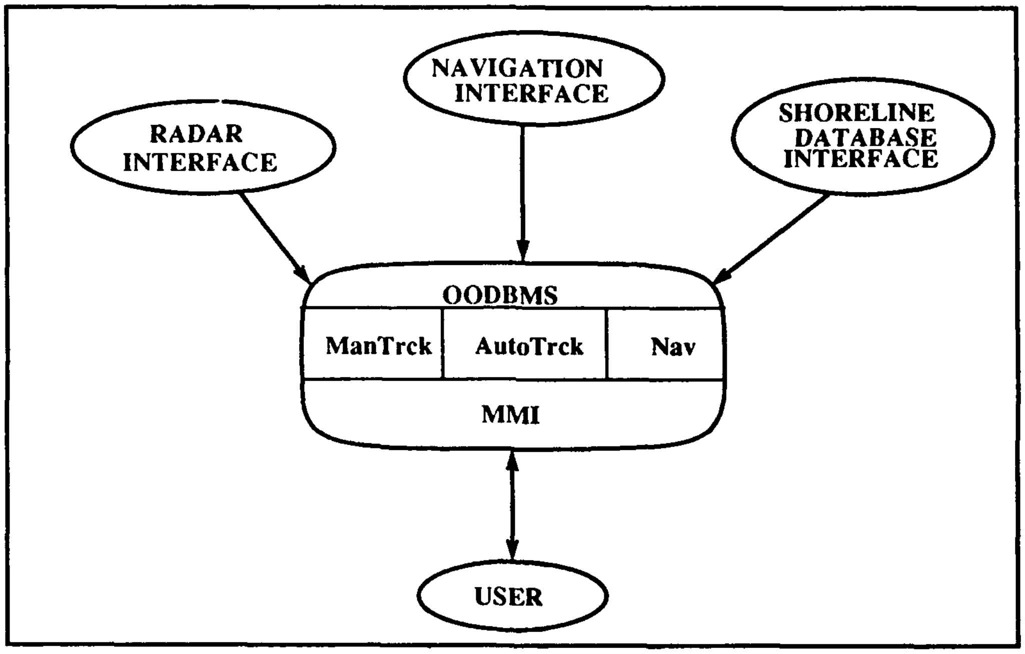


Figure 8 : Context Diagram for LCCDSWS, Increment I, II, III and IV

5. Increment Five

Integrate a receive only Link 11 capability which provides for the receipt and display of track information from the Ship to Ship digital data link. This increment will entail the development of an interface to a Link 11 system via NTDS protocol, parsing Link 11 messages and displaying parsed track information on the position display. It is not expected that the LCCDS will package and ship locally generated tracks for output on the data link. This will be a receive-only system. The specific format of the Link 11 data is specified in a classified Interface Design Specification (IDS) and

Operations Specification and will be provided under a different cover. For quote purposes assume twelve message types and six variations on each message type. NTDS interface boards are commercially available and will not require a hardware development effort.

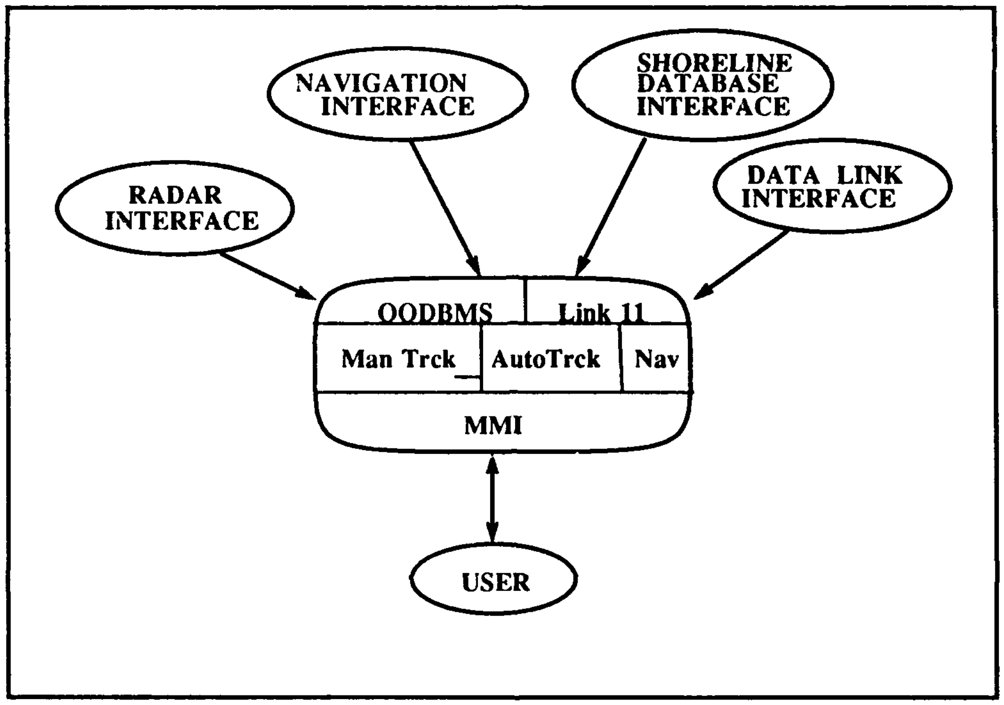


Figure 9 : Context Diagram for LCCDSWS, Increments I through V

C. ENVIRONMENT MODEL

1. Preface

The environment model defines the concepts needed to describe the world in which the proposed system will operate [Ref. 6]. The term "system" refers to the LCCDSWS in this context. The LCCDSWS shall be defined using the vocabulary of the existing CDSs and their environment as described in [Refs. 2, 7 and 9]. Figure 10 provides a view of LCCDS within the context of its external interfaces.

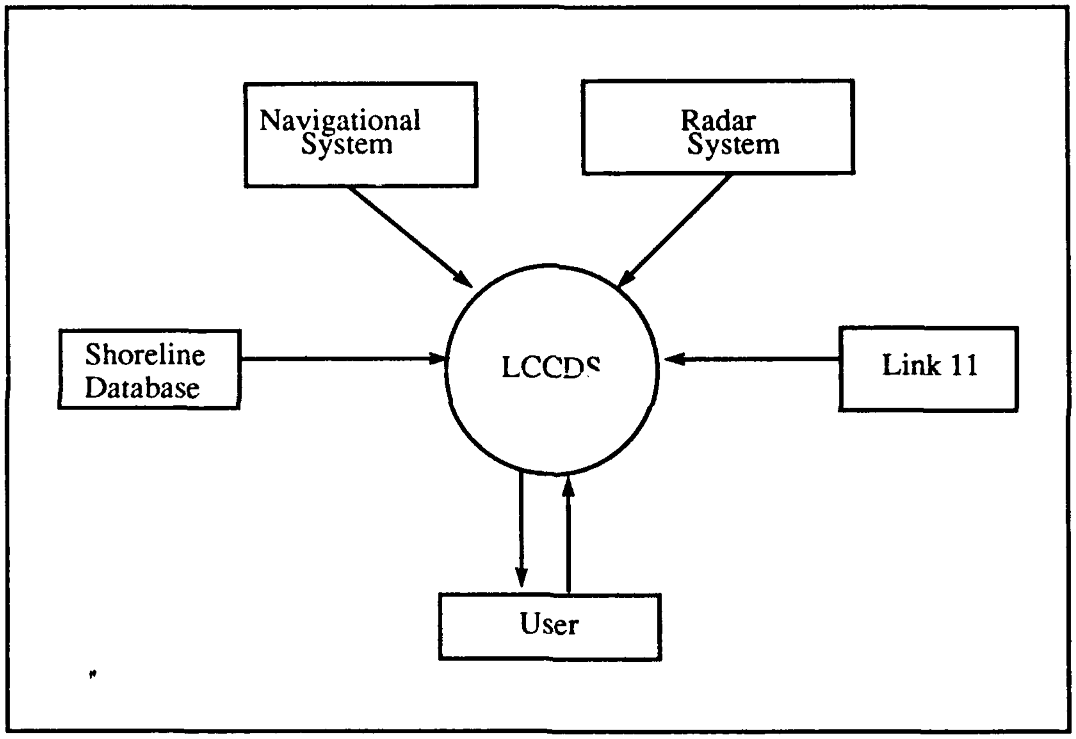


Figure 10: Initial Context Diagram for LCCDS

2. Terms and Definitions

- The ***LCCDS*** shall provide the user with a facility to monitor the overall air, surface and subsurface environment.

- The LCCDS shall collect, correlate and evaluate force and ownship sensor contact information.

- The ***LCCDSWS*** is the software system of the LCCDS.

- The LCCDSWS shall support a data display facility to implement the MMI described in increment one by providing a graphic presentation of graphic objects, icons, symbols and alphanumerics.

- The LCCDSWS shall be an automated shipboard database manager of tactically significant tracks.

- It will interact with the user, the tactical database, the Navigation System, the Radar System, the Link 11 System and the Shoreline Database.

- The ***Navigation System*** is a device that provides the system with ownship information in terms of present geographical position, course, speed and time.

- The ***Radar System*** is a device that provides the system with ownship sensor contact information in terms of bearing and distance.

- The ***Link 11 System*** is a device that provides the system with external information on tracks in terms of geographical position, course, speed, time, track number and track identity.

- The ***Shoreline Database*** is a device that provides the system with external information on shorelines.

- ***Geographical positions*** are positions relative to the earth's surface expressed in terms of latitude and longitude.

- A ***Relative Position*** is a position expressed in terms of range and bearing from an arbitrary point.

- *Course* is the angle between true north and the direction in which a ship submarine or aircraft proceeds.

- *True Bearing* is the angle between true north and an imaginary line between two objects.

- *Relative Bearing* is the angle between the bow of Ownship and an imaginary line between Ownship and another object.

- A *track* is a representation of some environmental phenomena converted into accurate estimates of geographical position with respect to time, course, speed, depth and height if applicable. A track is further described by a track number, track identity and category.

- A *vehicular track* is any object with attributes course and speed.

- *Dead Reckoning* is the estimation of a ship's position based on its course and speed and not from observation.

- *Track number* is a positive integer uniquely related to each track.

- *Track identity* classifies each track as one of the following: Friend, Hostile, Unknown and Special.

- *Track category* classifies each track as one of the following: Tentative, Air, Surface, Subsurface and Special.

- The *Closest Point of Approach* (CPA) is the position of a vehicle in range, true or relative bearing and time to another vehicle, when the two vehicles will be closest to each other.

- The *Data Link Reference Point* is a fixed geographic position representing the origin of a cartesian coordinate system in which track positions are reported.

- *Special Points* are a set of NTDS defined symbols, other than vehicular tracks, which supports the user in establishing a complete picture of the tactical environment. They may be fixed to a geographic point, or move with any valid course and speed.

- The ***Man-Machine Interface*** is that composition of hardware and software designed specifically to transform computer processed data into conceptual blocks of human understandable information, and act on the human response to this information.

- A ***user*** is any crewmember designated to operate the LCCDS for the purpose of providing the necessary information for quick, accurate tactical decision-making.

- A ***sensor*** is any device which somehow scans the external environment surrounding LCCDS and detects a certain class of tactically significant phenomena.

- The ***Hook*** function identifies a position on the TacPlot selected for some program action. Hooking an object is a standard NTDS user action and is described in further detail in G 1.2.1.1.2.

- A ***Tactical Plot*** (TacPlot) is a graphical representation of the spatial relationship between ownship and tactically significant tracks with respect to relative and geographic position.

- A ***TacPlot*** is a window for graphical input and output generated by the MMI.

- A ***symbol*** is the graphical representation/output on the TacPlot of an object within the object oriented database management system (OODBMS).

- A ***graphical input*** to the OODBMS is an input performed by the user on the TacPlot by selecting a position on the TacPlot with the Ball Tab.

- An ***object qualifies for graphical representation/display output*** on the TacPlot when its geographical position is in the set of geographical positions determined by the ownship geographical position and the range selected for that TacPlot.

- An ***alphanumerical window*** is a window for alphanumerical input and output generated by the MMI.

- ***Alphanumerical input and output*** to and from the OODBMS is the entry and retrieval of a set character strings associated with an object within the OODBMS.

- The attribute ***Origin*** describes the source of a track/special point object:

- ***Local Manual*** denotes an object who's source is user input.

- ***Local Auto*** denotes an object who's source is an ownship sensor.

- ***Remote*** denotes an object who's source is external, in particular Link 11.

D. LCCDSWS GOALS

To derive the high level goals we use the initial problem statement along with specific guidance as provided by NAVSEA [Ref. 4] and current CDS requirements documentation [Ref. 2, 7 and 9]. We assume that the reader is familiar with CDS terminology and the role and organization of the LCCDS.

G 1: Man-Machine Interfacc

The system has to provide a flexible, easy to use, window based user interface. We refer to this function as the *Man-Machine Interface (MMI) Function*. This goal will be implemented as part of Increment One.

G 1.1: Provide Tools for Interactive Operation

The system shall provide the user with the tools necessary to define his own customized screen formats for interactive operation of the system. This capability shall parallel current development for the Navy Command and Control Station (NCCS-A).

G 1.2: Provide Consistency with Standard Display

The system shall provide the basic features of a standard navy display console.

G 1.3: Provide a Display Doctrine

The system shall provide, via a *Display Doctrine*, the capability for user defined conditional statements, in IF-THEN form, which control data filtering and specify presentation parameters for displayed data.

G 2: System Control

The system has to control, initialize and monitor the LCCDS hardware, software and operator configurations. We refer to this function as the *System Control Function.*

G 3: Tactical Database

The system shall provide a Tactical Database. We refer to this function as the *Tactical Database Function.* This goal will be implemented as part of Increment One.

G 3.1: Integrate OODBMS into LCCDSWS

The Tactical Database shall be an object-oriented database management system (OODBMS). It shall be an integral part of the LCCDSWS.

G 3.2: Provide a Set of Object Types

The OODBMS shall provide a set of object types.

G 3.3: Allow Definition of new Object Types

The OODBMS shall allow the user to define new object types at runtime.

G 3.4: Provide Input and Output for OODBMS

The OODBMS shall provide features for data input and output.

G 3.5: Provide a Query Language for the OODBMS

The OODBMS shall provide a user friendly query language for building logical and arithmetic relationships between database objects.

G 4: Basic CDS Function

The system must support the user to establish an accurate picture of a ship's environment. We refer to this function as the *Basic CDS Function.* This goal will be implemented as Increments Two, Three, Four and Five.

G 4.1: Ownship Monitor Function

The system has to monitor the ships surface operations. We refer to this function as the *Ownship Monitor Function.* This goal will be implemented as Increment Three.

G 4.2: Tracking Function

The system has to update the position, determine course and speed and height and depth of applicable tracks. We refer to this function as the *Tracking Function.* This goal will be implemented as part of Increment Two.

G 4.3: Identification Function

The system has to support the determination of category and identity of applicable tracks. We refer to this function as the *Identification Function.* This goal will be implemented as part of Increment Two.

G 4.4: Search and Detection Function

The system has to accept tracks and sensor contact informations provided by ownship sensors. We refer to this function as the *Search and Detection Function.* This goal will be implemented as Increment Four.

G 4.5: Communication Function

The system shall support the establishment and maintenance of digital communications (Link 11 receive only). We refer to this function as the *Communication Function.*

G 4.6: Tableau System Function

The LCCDSWS shall provide detailed information on all aspects of the tactical situation and system control, operating parameters and status. This information is obtained from the Tactical Database, the user, external interfaces and various system parameters. It shall be indexed for easy user access and presented in tableau form using Alphanumeric I/O windows. We refer to this function as the *Tableau System Function.*

G 5: Incremental Development

The system shall be modularized to allow an implementation in incremental steps.

G 6: Concurrency and Extensions

The system shall be highly concurrent and prepared for future extensions.

E. LCCDSWS PROTOTYPE CONSTRAINTS

One of the primary goals of requirements analysis is to determine the constraints to be imposed on the proposed system. Constraints limit the range of choices available to the system developer and set the boundary conditions for system capability. A clear definition of constraints is essential to properly understand the trade-offs which will be necessary to resolve the inevitable conflicts with the stated goals of the system. The typical constraint types are Resource, Implementation and Performance [Ref. 6:p. 2-6]. Additional constraint types, regarding tactical data formats, units of measure and standardization are included to help define the environment in which the user must interpret the data presented. The following list of general, high-level constraints provides a brief perspective of LCCDS capability and limitations. These are described in greater detail in later sections:

C1: Resources

The level-of-effort NPS can provide the LCCDS program is limited only by the number of students interested in pursuing research related to the program.

C2: Implementation

The Low Cost Combat Direction Software System (LCCDSWS) prototype shall be implemented in Ada with an emphasis on use of commercial software, interfaced with Ada, for portable database and user interfaces.

C3: Performance

System response shall be sufficiently fast , and predictable under varying system loading to support critical decision making reliably in a highly stressful, rapidly changing tactical environment.

C4: Tactical Data Representation

The LCCDS Man-Machine Interface (MMI) shall represent all tactical symbology and data in a manner consistent with NTDS standardization and current fleet training conventions.

C5: Interface Compatibility and Standardization

The LCCDSWS shall provide the basic features of a CDS using existing standards as guidance for establishing a man-machine (MMI) interface "look and feel" which is consistent with other Navy command and control systems.

1. Resource Constraints

Resource constraints are administrative, rather than technical, in nature. They consist mainly of development schedules and milestones to be met and funding limits for the project. One of the major purposes of requirements analysis is to provide early warning when it appears that required goals and functions of the proposed system cannot be met given the time and money constraints.

At this time the LCCDS effort at the Naval Postgraduate School is in the form of unfunded thesis work conducted by graduate students. Thus, there is no formal schedule to be met, nor any budget concerns.

2. Implementation Constraints

Implementation constraints are normally imposed on the system designer by the customer. They concern hardware, operating systems, external systems (i.e., existing interfaces), programming languages and design standards.

a. Prototype Hardware

The Naval Postgraduate School has decided to use the Sun Microsystems Sparcstation 1 (RISC) machine for LCCDSWS prototype development. The selection of this particular hardware suite was based on the following criteria:

- The Sparcstation 1 meets the preliminary standards recommended for NGCR computers [Ref. 4].
- Ruggidized versions of the Sun workstations, capable of meeting military standards, are commercially available today [Ref. 10].
- Sun workstations are widely available within the Computer Technology Department, networked together via Ethernet.
- Students are familiar with the Sun systems.

b. Operating System

The Naval Postgraduate School will use the UNIX operating system as adapted by Sun Microsystems to support the use of windows for their workstations. The Sun operating system is derived from the UC Berkeley Version 4.2BSD and Bell Lab's UNIX System version 32V [Ref. 11:p. 10]. Use of a UNIX based operating system for prototype LCCDS software was based on the following criteria:

- UNIX based operating systems are well known and widely available to system developers.
- Software applications developed using UNIX are highly portable. There exists a great deal of information on how to design applications which will port easily to most UNIX derived operating systems [Ref. 12]. The software developed at NPS shall be directly usable on any UNIX based hardware suite.

The drawback in using UNIX for the prototype is that it does not provide mechanisms for guaranteeing real-time performance. There are, however several, UNIX-derived operating systems commercially available, which provide real-time scheduling capability and retain compatibility.

c. Implementation Language

In accordance with DoD policy and as specified in the NAVSEA Statement of Work (Appendix A), Ada shall be used as the implementation language for LCCDSWS.

d. Commercial Software

As recommended by the NGCR [Ref. 3], maximum use of existing commercial software shall be utilized in the development of the LCCDS prototype. Specifically, use of commercial window and database management software shall be explored.

e. External Interfaces

As specified by NAVSEA [Appendix A], the LCCDS shall eventually be capable of interfacing with existing shipboard navigation, radar and Link 11 equipment. These can be viewed as "passive" interfaces in that LCCDS will receive data from, but not provide control of, these external systems.

The database systems interfaces and the user interface shall be considered as part of the LCCDSWS. As stated in subparagraph d above, the database and windowing capabilities shall take advantage of available commercial software to the maximum extent possible.

f. Detailed Implementation Constraints

C2:

The Low Cost Combat Direction Software System (LCCDSWS) prototype shall be implemented in Ada with an emphasis on portability and use of commercial software interfaced with Ada, for database and user interfaces.

C2.1:

The 4.2BSD UNIX operating system, as implemented in the Sun Microsystems Sparcstation 1, shall provide the LCCDSWS run-time environment and host hardware for prototype development.

C2.2:

The user interface shall be designed using current windowing technology available on microprocessor based workstations. The use of commercial window management software, interfaced with Ada, is emphasized.

C2.3

The Tactical Database shall be object-oriented, allowing all associated track data (text, graphics, functions) to be stored as a single entity. The database shall be designed to utilize dynamically allocated storage. Maximum number of stored tracks shall be limited only by physical memory availability.

C2.4

The prototype LCCDSWS shall be developed under the assumption that the host environment will be different from the development environment. Therefore, the system must be portable, supporting maximum independence from development hardware.

3. Performance Constraints

Performance constraints may be viewed from several different levels. The LCCDS is primarily an interactive system and therefore must provide fast, predictable response to operator demands. At the system level, performance constraints specify maximum program size (memory space required), reliability and performance under various conditions of system degradation (e.g., casualty modes of operation available).

At a lower (functional) level, performance constraints are more focused. The LCCDS will play a major role in the tactical decision making process, thus requiring hard real-time response for some functions. While the prototype may not meet real-time reliability constraints, as explained in the previous section, the software will be developed as a real-time system with specific timing constraints associated with critical functions.

a. Real-Time Performance

For the purposes of the LCCDS real-time performance means providing *guaranteed* response to external input and user demands within the specified timing constraints. The LCCDS shall be viewed as a *deadline driven system* where late information may be useless and result in severe or catastrophic consequences.

b. Detailed Performance Constraints

C3:

System response shall be sufficiently fast, and predictable under varying system loading to support critical decision making reliably in a highly stressful, rapidly changing tactical environment.

C3.1:

All tracks shall be updated at least every 4 seconds.

C3.2:

Response times for menu selections, track information requests, tactical display aids (graphic tools), etc., shall be less than 1 second.

C3.3:

The system shall support up to 1000 active tracks/special points displayed on the TacPlot. The number of tracks/special points held in the Tactical database shall be constrained only by physical memory limits of the system.

C3.4:

Display of shoreline maps on the TacPlot is not subject to real-time performance constraints.

4. Tactical Data Representation Constraints

Display constraints are imposed on the representation of tactical symbol and data as necessary to support standardized training, operations and communication between CDS configurations. Detailed information and guidance regarding standard display symbology for combat direction systems is contained in the FCDSSA CDS Standardization Manual [Ref. 13].

C4:

The LCCDS MMI shall represent all tactical symbology and data in a manner consistent with NTDS standardization and current fleet training conventions.

C4.1:

The MMI shall present the tactical picture and associated data in a manner consistent with existing combat direction systems.

C4.1.1:

All range information shall be expressed in nautical miles (NM) and/or yards (yds).

C 4.1.2:

All azimuth (bearing/heading) information shall be expressed in degrees (000.00-359.99) relative to either true or magnetic north as indicated by an uppercase T or M immediately following second decimal place of the number.

C 4.1.3:

Altitude and depth information shall be expressed in feet (ft).

C 4.1.4:

Time shall be expressed in 24-hour clock digital format followed by the appropriate upper case letter time zone identifier. The default time zone shall be Greenwich Mean Time (GMT). The user shall have the option of displaying current time in any time zone.

C,4.1.5:

Geographic position shall be expressed in terms of latitude (north-south) and longitude (east-west) degrees, minutes and seconds.

C 4.1.6:

Speed shall be expressed in knots (kts).

C4.2:

All LCCDS terms, acronyms and data naming conventions shall be limited to those commonly used and meaningful within the NTDS environment.

5. MMI Compatibility Constraints

The LCCDS MMI, to the maximum extent possible, shall follow existing standards in regard to MMI design for graphics workstation applications. The most current standard available is the SPAWAR Software Requirements Specification for the NCCS-A Workstation Executive [Ref. 5]. This detailed specification provides extensive guidelines and design approaches for MMI development taking advantage of current state-of-the-art display technology. The obvious course of action, therefore, would be to constrain the development of LCCDS to conform as much as possible to the MMI conventions specified for NCCS-A, and fitting it to CDS function.

C5:

> The LCCDSWS shall provide the basic features of a CDS using existing standards as guidance for establishing a man-machine (MMI) interface "look and feel" which is consistent with other Navy command and control systems.

C5.1:

> The LCCDS MMI development shall be constrained to follow the form and environment described in Reference 5 to the maximum extent possible while preserving the functional goals of a CDS as described in this document.

F. LCCDSWS GOAL/FUNCTION REFINEMENT

1. Man-Machine Interface (MMI) Function

a. Preface

The Man-Machine Interface is the largest and most complex portion of the LCCDSWS. It is here that the machine *touches* the user. All system control, function and capability shall be accessed via this MMI. The interface is primarily visual, providing the user with graphic representation of tactical data, spatial relationships and tactical decision aids using window based menus, cues, tableaus and graphical plots.

b. Goals/Functions

The following system goal statements reiterate the high level requirements and refine the functional detail of the MMI.

G 1: The Man-Machine Interface

The system shall provide a flexible, easy to use, window based user interface. We refer to this function as the Man-Machine Interface (MMI). The MMI shall consist of a Tactical Display, trackball, keyboard and associated software necessary for control and manipulation of the tactical data presentation.

The Tactical Display shall consist of a set of active windows displayed on a 19 inch bit-mapped color monitor. User control and data I/O to the LCCDS shall be via the MMI.

The MMI shall provide the user with tactical information, defined and filtered as specified by a Display Doctrine, and the means to control and modify the system and tactical information presented. Figure 11 provides a detailed depiction of MMI structure and components.

G 1.1: MMI Building Blocks

The system shall provide the tools necessary to define customized versions of the Tactical Display screen for interactive operation of the system.

G 1.1.1: Tactical Display Architecture

Maximum use of current windowing technology shall be used to implement the Tactical Display portion of the MMI.

Windows are rectangular areas positioned on the display screen in a manner allowing the user to arrange space and information to best meet his needs.

Windows shall be allowed to overlap or nest inside other windows.

Obscured windows shall retain their I/O functions and capability.

Windows shall be mapped to any position on the Tactical Display as selected by the user.

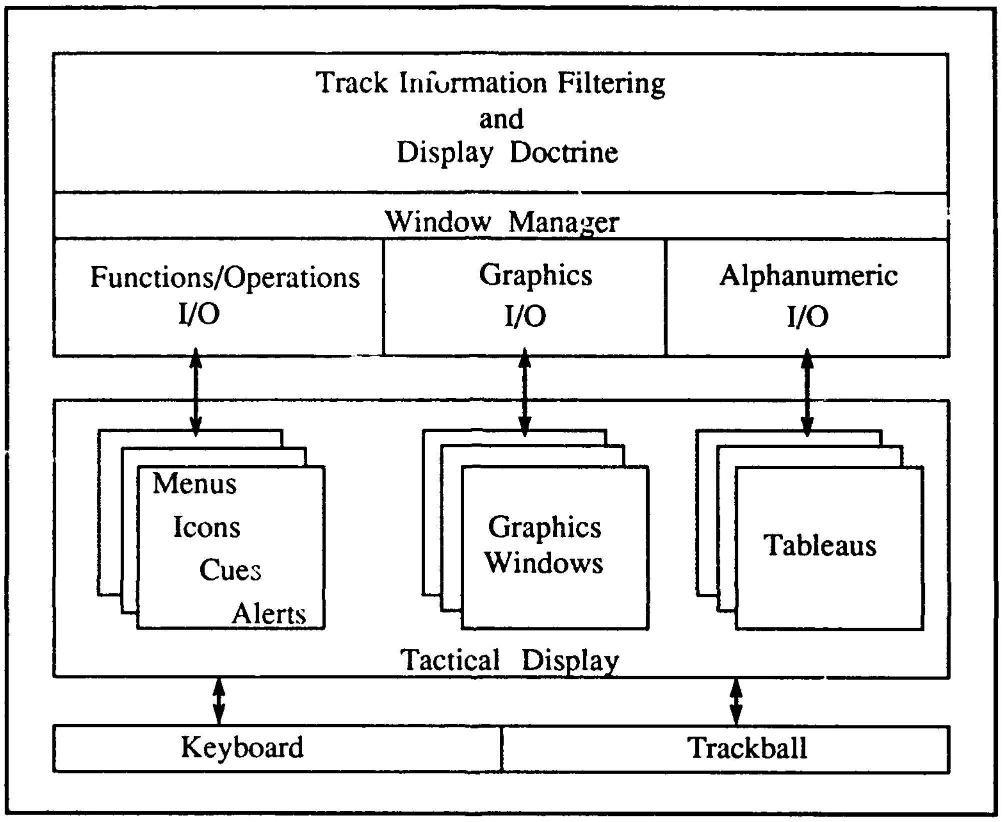


Figure 11 : Man-Machine Interface (MMI) Structure

Each window shall have a stored set of attributes, referred to as it's *Graphic Context* (GC) [Ref. 14:p. 39], which specify the following:

- background color
- foreground color (text, graphic objects, icons)
- window position (current mapping)
- window exposure state (exposed or hidden)
- window size (fixed or variable)
- border style

- cursor type (ball tab or arrow pointer)
- I/O capability

Each window GC shall contain a set of predefined default values and be user modifiable.

User choices for GC attributes shall be limited to remain within the constraints set by US Navy human factors guidance.

G 1.1.1.1: Window Classes

There shall be three basic classes of window each providing the user with a particular method for performing I/O; Functions I/O, Graphics I/O and Alphanumeric I/O. The visual appearance and functionality of these window classes shall be implemented in accordance with the conventions described in Goal 1.1.1.2.

G 1.1.1.1.1: Functions I/O

The Functions class includes pop-up menus, icons, cues and alerts which provide the user input, information and/or feedback on all LCCDSWS software functions and operations.

The user performs the necessary *I/O using Trackball cursor* (Arrow Pointer) and Hook functions to select and activate menu choices or icons associated with the desired system function.

Any function requiring the user to complete a sequence of actions shall provide menu driven choices in the form of pop-up/pull-down windows. The user shall be able to directly access the window for data input, modification and/or menu selections via keyboard or cursor. *System reliance on user memory shall be avoided to the maximum extent possible.*

Those functions which are frequently used, repetitive in nature, or consist of only one step or choice of action may be represented in icon form.

Visual cues and alerts signalling for user action, or in response to user actions or requests shall be generated in the form of pop-up windows.

Any user request or input to the system shall be graphically acknowledged via text windows, highlighted menu choices, inverse video, change in cursor shape or color, etc. The system shall provide timely feedback, in some form, to assure the user of pending program action or response.

G 1.1.1.1.2: Graphics I/O

The Graphics I/O class provides for Plan Position Indicator (PPI) like display of tactical data and symbology in accurate spatial relationship with each other and Ownship position.

The primary instance of this window class is the TacPlot.

User control and manipulation of tactical data displayed on the TacPlot shall be via Trackball cursor (Ball Tab) and Hook functions.

G 1.1.1.1.3: Alphanumeric I/O

This class provides alphanumeric information, in tableau form, regarding detailed tactical data for Ownship, all tracks held in the Tactical Database, system status and operating parameters, display configuration, window GC's.

This class shall provide the user direct access to the Tactical Database for information and manipulation of tactical data. The user performs the necessary alphanumeric input into data tableaus using the keyboard.

G 1.1.1.2: Basic Conventions for the MMI

To aid in the standardization effort, the conventions for the MMI shall be consistent with the NCCS-A Workstation Executive requirements [Ref. 5]. The following is a goal refinement for the LCCDS MMI as derived from that specification.

G 1.1.1.2.1: User Interaction Techniques

Normal user interaction with the system will be via the trackball (or mouse if employed) and keyboard. All cursor interaction shall be via a three-button trackball or mouse, or the cursor keys (hereafter referred to as the trackball). The trackball controls a cursor which, when interacting with menus, windows or objects (such as tracks), is displayed as an arrow known as the pointer. The actual shape of the cursor shall be redefined by an application for use within the confines of its own windows only. In a text window, the cursor will be in form of a vertical blinking bar, in a TacPlot window the cursor will be in form of the ball tab (Goal 1.2.1.1).

As the trackball is moved, the cursor will follow on a corresponding path across the screen. Certain trackball movements, when combined with trackball button interaction, are used for system control.

G 1.1.1.2.1.1: Trackball Interaction Terminology

This goal describes the terminology required to describe trackball interaction:

(1) **Press:**

Depress a trackball button and hold it down.

(2) **Release:**

Release the appropriate trackball button to initiate the desired action.

(3) **Click:**

When the cursor is located over a position or object of interest on the screen, a quick press and release of the trackball button creates a signal recognized by the system as a "click". Clicking has differing effects depending on the cursor's location. In general clicking the trackball is equivalent to pressing the return key on the keyboard. When in a TacPlot, an object may be selected by clicking the trackball button once. Clicking the object a second time deselects it.

(4) **Moving the Cursor:**

Rolling the cursor with no buttons depressed or activated shall move the cursor in the corresponding direction.

(5) **Dragging the cursor:**

This technique is used to move or reconfigure an object on the screen. Dragging is performed by placing the cursor over the appropriate object, clicking a trackball button and then moving the cursor. When the desired effect is accomplished, the user clicks the trackball button again and the system reconfigures the screen image as appropriate.

G 1.1.1.2.1.2: Basic Interactive Window Features

All interactive windows for Function-, Graphics- and Alphanumeric I/O shall be represented using the following conventions in accordance with Reference 5.

G 1.1.1.2.1.2.1: Basic Window Interaction

Interactive windows shall be designed using the basic building blocks as described in the following paragraphs. These window features are depicted in Figure 12.

(1) **Close Box**:

The Close Box is a small square icon which, when located in the left-hand corner of the title bar, shall be used to close the application window. Clicking on it means no further processing by this application window is required and to close the window. If the user has performed any editing or changed any settings, the user will be asked whether he wishes to save the changes made.

(2) **Iconify Box**:

This box is a small square object resembling a four-pane window located on the right end of the title bar. When the user clicks the iconify box, the window shall shrink to icon size and move to the upper left-hand side of the root window. This feature shall only be used in applications which either require user initiation and then can run in the background, or may be interrupted to perform other processing. The user shall be able to move the application icon.

(3) **Resize Box**:

This box is a small icon consisting of three squares of decreasing size which, when located in the lower right corner of the application window, may be used to resize the window. It shall be activated and used by dragging the cursor. As the cursor is moved, it shall drag a dynamically adjusting outline of the window with it. The outline shall contract if the cursor is moved toward the upper left-hand of the window and expand if the cursor is moved away. The window shall remain in the desired size as soon as the trackball button is clicked a second time.

(4) **Help Box**:

This box is a small square icon containing a question mark. It will normally be positioned in the right corner of the title bar, and just to the left of the iconify box if present. When the cursor is placed over the help box and the trackball button is clicked an alphanumerical, scrollable window will be displayed which provides access to help for the application window.

(5) **Title Bar**:

A rectangular bar at the top of the window which includes the title of the window (and the window number if multiple copies are allowed), and may contain amplifying information from the application creating the window, a close box, help box and or/an iconify box. If a window title is bracketed on both sides, or at least the right side by a group of horizontal lines denoting a drag bar, the window may be repositioned on the screen within the confines of its parent windows.

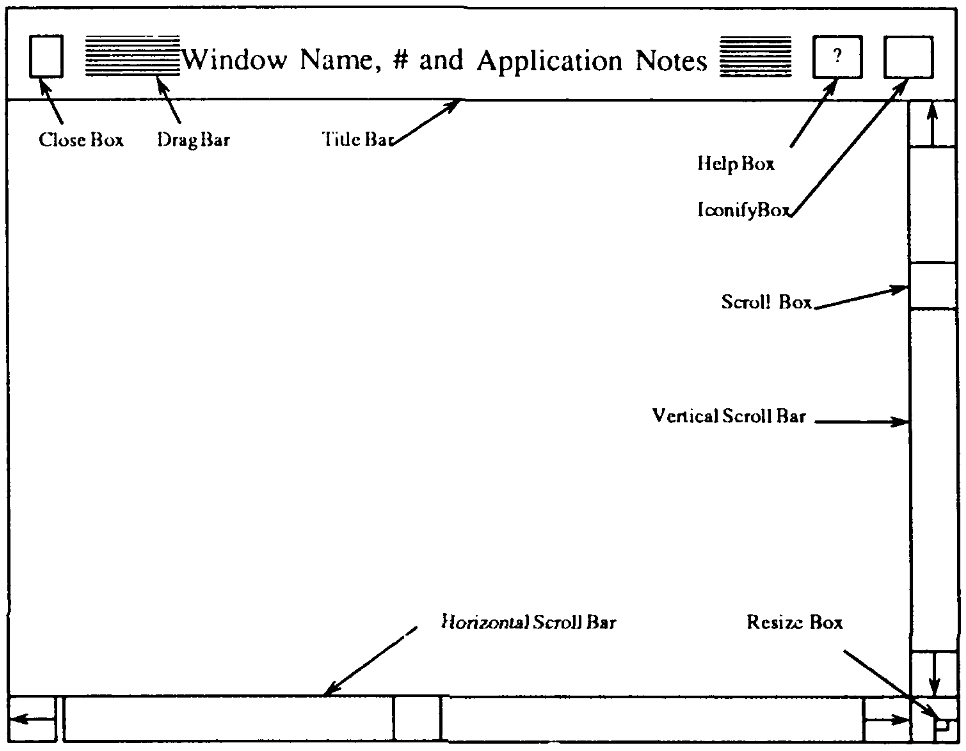


Figure 12 : Basic Interactive Window Format

(6) **Scroll Bars**:

Scroll bars are used to change the view of the contents in a alphanumerical or graphical window which does not completely fit within the confines of the window displayed. A vertical scroll bar on the right-hand side of the window supports paging forward and backward through a document or text while a horizontal scroll bar on the bottom edge of the window provides horizontal movement.

In a graphical window the vertical scroll bar supports South-North panning and the horizontal, East-West panning. The components of a scroll bar are the scroll arrows (located at the end of each scroll bar), a white box in a shaded area and the shaded area. Clicking a scroll arrow causes the window to scroll one line (or column as appropriate) in the direction of the arrow (in a graphics window, one-eight of the viewable page). The down arrow shall provide scrolling toward the end of the file, and the up arrow toward the beginning of the window content. Pressing and holding on an arrow causes continuous scroll until the button is released or the end of the window content is reached. Clicking in the area on either side of the scroll box causes the box to move up or one page in the direction indicated. The scroll box moves in the direction clicked a distance proportional to the relative movement through the window content. If the user places the cursor over the scroll box and drags the cursor along the shaded bar, the scroll box will follow. When the trackball button is clicked a second time, the window content will scroll to the new relative position of the scroll box on the shaded bar. If an entire scroll bar is white, then the entire document or graphic area is completely visible in its direction.

(7) **Push Buttons**:

Shall be displayed as a rectangular object with rounded corners and a label in the center. When clicked, they shall cause the application to execute the captioned command. A yellow push-button shall indicate a recommended action; all other push-button shall be red.

(8) **Check Box**:

A check box is essentially an On/Off switch. On when the small square box contains a check mark and Off when the box is empty. Clicking on an On check box shall turn it Off and vice versa.

(9) **Radio Buttons**:

May be used when selecting from multiple options where only one can be selected. When one radio button is selected, the previously selected button is unselected. Clicking on the selected button leaves it selected. An empty circle within a circle icon shall denote that the item is not selected, and a black filled circle within a circle icon shall indicate that it is selected. As shown in Figure 13, radio buttons that refer to similar kinds of options should be grouped in sets and arranged in columns or rows. A label that describes the choice represented by the radio button shall be displayed to the right of the button.

G 1.1.1.2.1.2.2: Window Manipulation

Through the use of the trackball, the user shall be able to perform the following manipulations on windows which include a drag bar and a resize box:

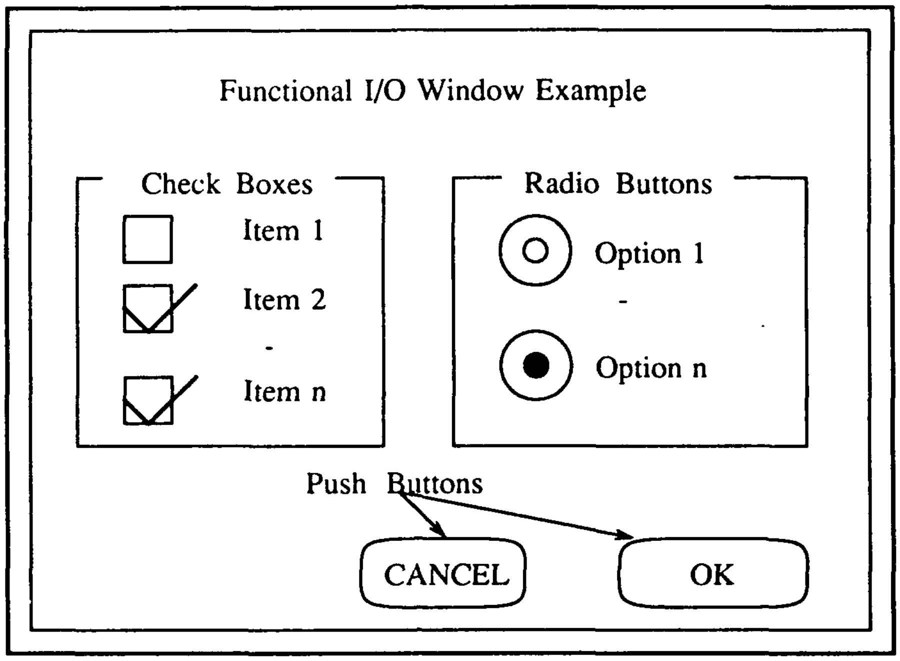


Figure 13: Standard Support Window with Check Boxes, Radio and Push Buttons

(1) **Place**:

A window shall be placed initially at a position and size set by the application. The user will then be able to move the window to a desired position and change its size as described below. The user shall be allowed to save the new position and size of the window.

(2) **Move**:

Windows may be moved by placing the cursor on the drag bar, clicking the trackball button, moving the cursor to the desired position, and then clicking the trackball button again. When the button is initially clicked on the drag bar, a window frame shall appear and move with the cursor until the next trackball click. When the trackball button is clicked the second time, the window shall be redrawn in its new position.

(3) **Resize**:

A resize on the window frame shall be performed by placing the cursor on a corner side of the window and dragging. The procedure to resize a window using the resize box is described in Resize Box (Goal 1.1.1.2.1.2.1).

G 1.1.1.2.1.2.3: **Window States**

(1) **Input Focus**:

The "Input Focus" is the window that currently receives keyboard and trackball inputs. By default the focus is the Root Window (Goal 1.1.1.3.1). When multiple windows are displayed, the operator shall be able to designate a window as the new focus by clicking into it. Once designated as input focus, the window shall brought to the front. However it shall not be possible to place other windows over warning windows or the main menu bar. When the user picks an option and a window is opened, the focus may shift to that new window.

(2) **Active**:

A window is active when the associated processing continues irrespective of which window has the input focus. A TacPlot window shall always be active unless closed by the user. The clock window shall always be active.

(3) **Open**:

Multiple windows may be open on the display at one time. The input focus window is active until the function is completed, the operator closes the window, changes the input focus to another window or activates another window. The most recently activated window shall come to the front.

(4) **Icon:**

An active window which does not require user interaction for extended periods can be run as a background task. To serve as an indicator that the application is running and to provide quick access when changes are required, such windows may be iconified using the Iconify Box as described above.

(5) **Close**:

When a window is closed, it is removed from the appropriate display and all processing relating to it is stopped.

G 1.1.1.2.1.2.4: **Window Formats**

All windows shall adhere to one of the following formats:

(1) **Plain Box Window**:

A plain black border box with no close, help, iconify or resize boxes (used for the clock and menus). These windows are application controlled and may not be moved by the user.

(2) **Graphics Application Window**:

One example is pictured in Figure 14. It includes a black border box with a graphics application menu bar just below the title bar across the top of the window. Graphics windows may be scrollable as appropriate for each individual application. The bottom left-hand side may be available for application icons or alphanumeric remarks.

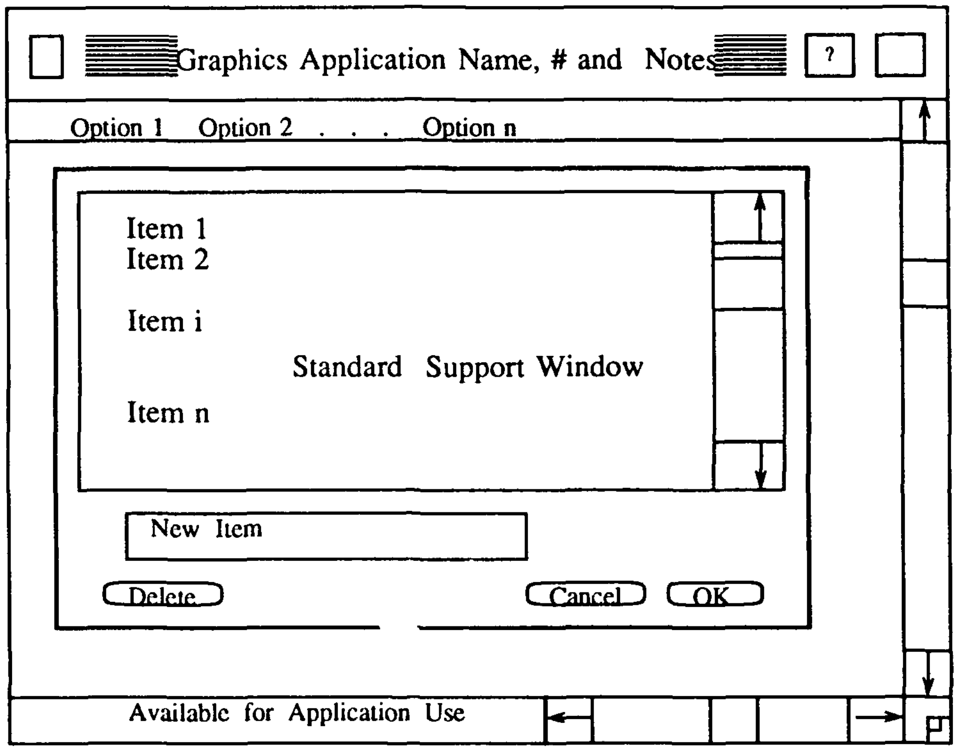

Graphics Application Name, # and Notes
?
Option 1 Option 2 . . . Option n
Item 1
Item 2
Item i
Standard Support Window
Item n
New Item
Delete
Cancel
OK
Available for Application Use

Figure 14 : Graphics Application Window and Support Window

(3) **Scrollable Document Window**:

This window is pictured in Figure 15. It shall consist of a black border box with a title bar, vertical and horizontal scroll bars as appropriate and an area for use by the application in the bottom left-hand corner. The window may be dragged to a new location, resized or iconified if appropriate.

(4) **Standard Document Window**:

This window is pictured in Figure 20 and shall consist of the same features as described for the scrollable document window except for the scroll bars. It may be dragged, resized or iconified if appropriate.

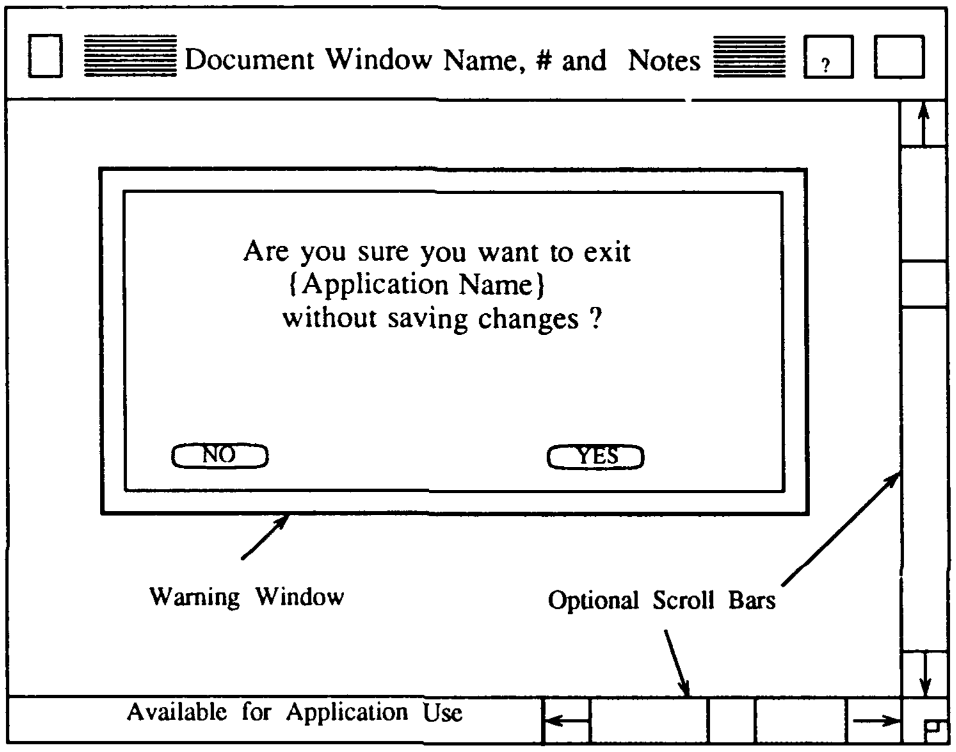

Document Window Name, # and Notes
?
Are you sure you want to exit {Application Name} without saving changes ?
NO
YES
Warning Window
Optional Scroll Bars
Available for Application Use

Figure 15: Document Window with Scroll Bars and Warning Window

(5) **Standard Support Window**:

This is a fixed-location window which requires an operator response to close. It has a double border and normally will include at least two response buttons , one to indicate the information is "OK" as currently set to execute the task and close the window, and a second to "CANCEL" the window with no action taken even if the user has entered information. Figure 13 shows an example of a Standard Support window which uses Check boxes, Radio and Push Buttons. Figure 14 shows an application using scroll bars.

(6) **Moveable Support Window**:

This is a Standard Support Window with a title bar and drag bar functionality. Figure 18 depicts an application using a Moveable Support Window.

(7) **Warning Windows**:

These are used to alert the user to an abnormal condition. It is a special version of the standard support window and shall at least include "Yes"/"No" buttons and the information necessary for the user to act accordingly.

G 1.1.1.2.1.2.5.: Menus

A menu is a list of program options a user can choose from. The approach to be implemented in LCCDSWS has to be consistent with [Ref. 5, p.25]. In LCCDSWS at least the Tactical Display Main Menu and the Tactical Plot Menu Manager will be visible. Primary method of interaction is via trackball.

(1) **Main Menu Bar**:

In the second row of the Tactical Display Screen (root window) is a one-line bar with the names of the various menu options as illustrated in Figure 16. Normally, each menu option will have one level of sub-menus displayable as a pull-down menu. Menu options requiring additional information from the operator to execute shall normally use support windows and support menus. While an operator is interacting with the application associated with a menu option, that menu option is displayed in reverse video. This includes the occasions when a sub-menu is displayed and when an application window is displayed as an icon.

(2) **Pull Down Menu**:

Implement in accordance with Reference 5, Paragraph 3.2.1.1.6.2.

(3) **Application Window Menu Bar**:

Implement in accordance with Reference 5, Paragraph 3.2.1.1.6.3.

(4) **Support Menu**:

Implement in accordance with Reference 5, Paragraph 3.2.1.1.6.4.

(5) **Menu Formats and Features**:

Implement in accordance with Reference 5, Paragraph 3.2.1.1.6.5.

G 1.1.1.2.1.2.6: Window Layout Requirement

The window layout requirements shall be consistent with the layout requirements for NCCS-A windows as described in Reference 5, Paragraph 3.2.1.1.7. This includes the color scheming.

G 1.1.1.2.1.2.7: Keyboard Guidelines

All LCCDS workstation keyboards shall meet the requirements for NCCS-A workstation keyboards as described in Reference 5, Paragraph 3.2.1.1.8.

G 1.1.1.2.1.2.8: Help Guidelines

The general characteristics for the "HELP" function in LCCDS shall meet the requirements for NCCS-A as described in Reference 5, Paragraph 3.2.1.1.9.

G 1.1.1.2.1.2.9: General Functionality Guidelines

General functions are used to edit graphics and text in multiple windows. They shall be consistent with Reference 5, Paragraph 3.2.1.1.10.

G 1.1.1.3: LCCDS Windows

This following shall provide concrete examples on the appearance and interaction features of some of the basic and most important windows to be used in LCCDS. It shall be understood that these requirements are not static, rather then giving a concept how the above conventions shall be applied to the windows used. It is expected that appearance and interaction techniques for this project will mature as experience is gained through implementation and analysis of the prototype. The main purpose of this goal is to provide examples how MMI features for LCCDS shall be constructed from the building elements of Goal 1.1.1.2.

G 1.1.1.3.1: The Root Window

The root window provides a fixed background for the Tactical Display and shall cover the whole screen. It is not moveable or sizeable and represents the basic workbench for the user. All other windows will appear overlaid over the root window, no window can be hidden behind the root window. However, the title bar and the main menu bar of the root window shall always be visible. The root window is pictured in Figure 16.

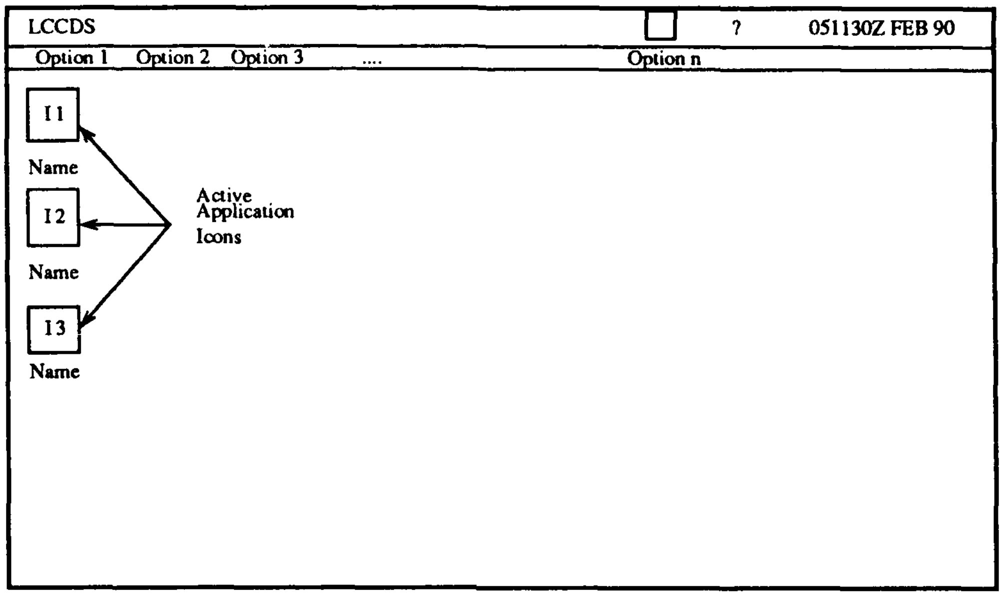


Figure 16 : LCCDS Root Window

G 1.1.1.3.2: The Tactical Plot Window

The TacPlot windows will be a Graphics Application Windows as described in G 1.1.1.2.1.2.4 and used for graphics I/O. Their possible appearance is shown in Figure 17. The figure shows some examples of the standard NTDS symbology as described in Goal 1.2.1.1.

G 1.1.1.3.3: The Radar and Video Select Window

This window in figure 18 provides Functional I/O for selecting a radarvideo of an external Radar-System. It is built from a Moveable Support Window together with Radio and Push Buttons. Note that it can be iconified.

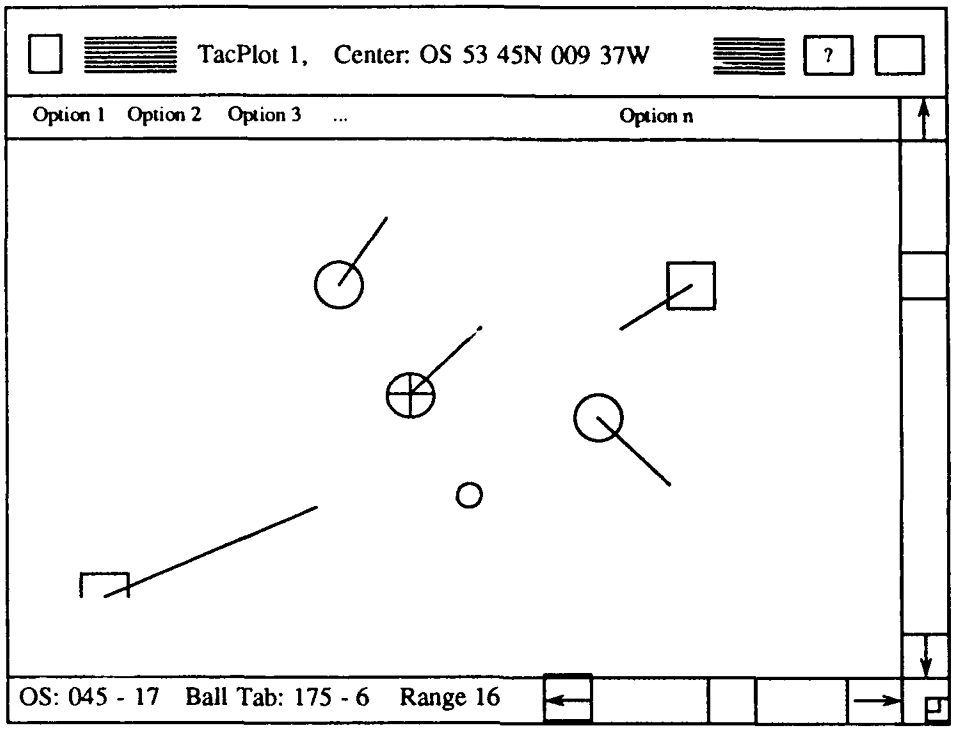

TacPlot 1, Center: OS 53 45N 009 37W
?
Option 1 Option 2 Option 3 ... Option n
OS: 045 - 17 Ball Tab: 175 - 6 Range 16

Figure 17 : The Tactical Plot Window

G 1.1.1.3.4: The Interface Control Window

The Interface Control Window is another example of Functional I/O. It is used to establish communication channels to external subsystems. It is constructed using support window, dragbar, iconify box, check boxes and push buttons. An example is shown in Figure 19.

G 1.1.1.3.5: New Track Establishment Window

Figure 20 depicts an example of an Alphanumerical I/O window, in this case the manual entry of a new track into the system. It uses a standard document window and push buttons to confirm or cancel the entries and provides a capability to display and enter data manually in support of Goal 3.2.1. The window could be used as result of a grap[illegible] I/O in the TacPlot to manually enter a new track

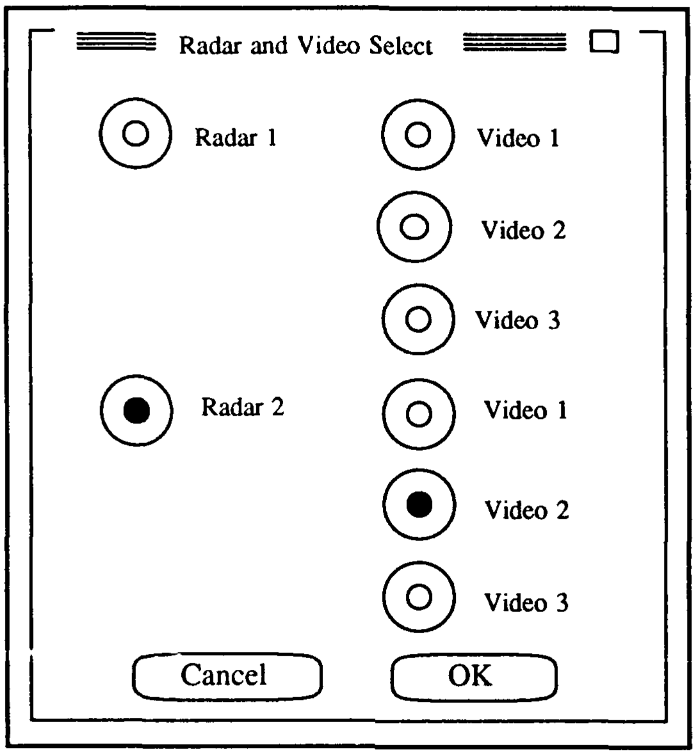


Figure 18 : Radar and Video Select Window

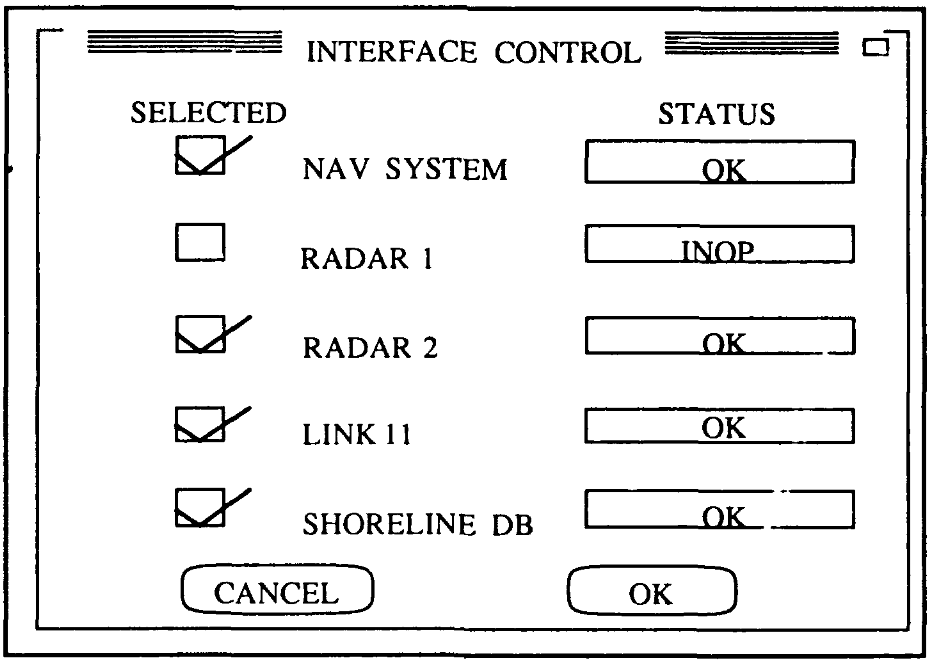
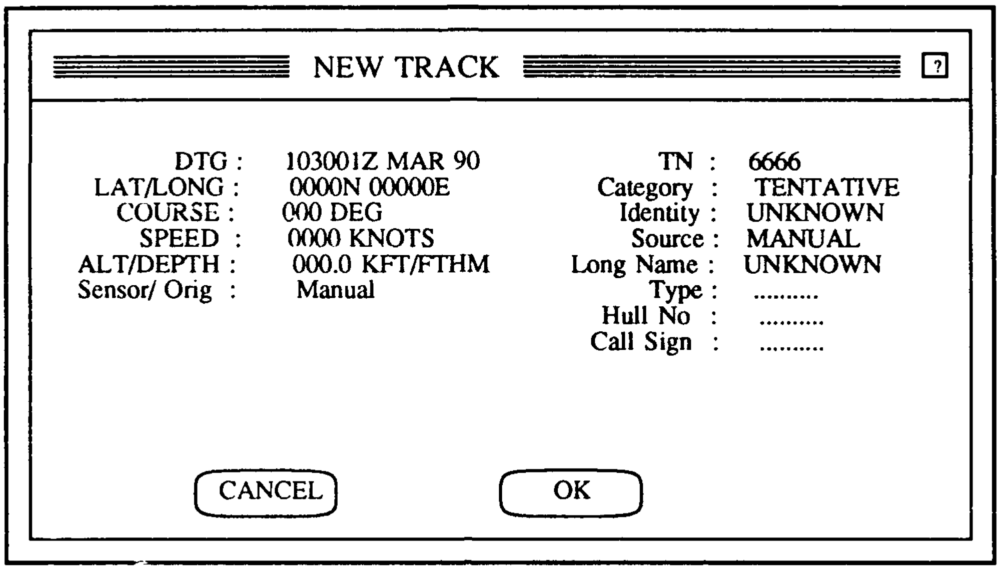


Figure 19 : Example Function I/O Window



DTG :	103001Z MAR 90	TN :	6666
LAT/LONG :	0000N 00000E	Category :	TENTATIVE
COURSE :	000 DEG	Identity :	UNKNOWN
SPEED :	0000 KNOTS	Source :	MANUAL
ALT/DEPTH :	000.0 KFT/FTHM	Long Name :	UNKNOWN
Sensor/ Orig :	Manual	Type :	....
		Hull No :	....
		Call Sign :	....

Figure 20: Standard Document (New Track Establishment) Window

G 1.2: Tactical Display Features

The system shall provide the basic features of a standard Navy display console.

The Tactical Display shall consist of a tactical plot (TacPlot) and associated support functions, menus, user cues and alerts. A moving cursor display (Ball Tab or Arrow Pointer) shall be available to allow the user to designate any position on the Tactical Display for control, information and/or data input to the system.

The Cursor symbol shall be controllable by either trackball or keyboard.

The Ball Tab shall represent the Cursor display within the TacPlot Graphics window class. Any time the cursor is positioned outside the boundary of this window class it shall be represented as an Arrow Pointer.

The system shall provide a facility for fixed and variable (operator defined) function keys. The function keys shall provide for control, data entry and interaction with the software system.

G 1.2.1: Tactical Plot (TacPlot)

The tactical plot shall display data denoting tactical information pertaining to an area of surveillance.

It shall be capable of displaying up to 1000 symbols (Tracks and Special Points).

The TacPlot shall be based on a X-Y coordinate system with origin at Ownship position.

The TacPlot shall provide geographic information (latitude/longitude) on any selected (hooked) position.

The system shall provide for selection of range scales.

The system shall allow the user to display a proportional speed leader on each track.

The system shall allow the user to filter out undesired symbology.

G 1.2.1.1: NTDS Standard Symbology

TacPlot functions and symbology must be consistent with NTDS definitions as specified in References 7 and 13. The system will provide the following TacPlot functions and associated symbology extracted from Enclosure 1 to Reference 4 (see Appendix A). These symbols shall be kept separate from other (system control) symbology and will not appear outside of the window defining the TacPlot:

(1) Ball Tab: ⊙

The Ball Tab is a moving cursor represented by a one-quarter inch diameter circle with a point in the center. The Ball Tab shall provide direct feedback to the operator as he manipulates the trackball. Movement of the Ball Tab within the TacPlot shall match the relative speed and direction of any trackball movement.

There shall be only one Ball Tab symbol displayed on the TacPlot at any time.

(2) Hook: ○

The Hook function shall identify a position on the TacPlot selected for some program action. As shown above the Hook symbol is represented as a one-half inch diameter circle.

The Hook function shall operate in conjunction with the current position of the Ball Tab. Any point within the TacPlot may be identified by positioning the Ball Tab and actuating the Hook function by clicking on a trackball button.

In the case where the position "hooked" coincides with a "hookable" symbol, the Hook symbol shall appear surrounding that symbol. Only one symbol at a time may be hooked. The following symbols shall be "hookable":

- Tracks
- Ownship
- Waypoints
- Man in Water
- Navigation Hazard
- Data Link Reference Point
- Reference Points
- Utility Lines and Circles

If there is no symbol located at the position identified by the Ball Tab, then that position shall be saved for further program action. *In this case the Hook symbol shall not appear.*

(3) Track History Point: •

The functional aspect of this symbology is explained in Goal 4.2.5.

(4) Reference Points: X

A Reference Point function shall allow the user to display up to 20 Reference Point symbols at any hooked position on the TacPlot.

(5) Tracks:

Tracks are symbolic representations of remote and locally generated real world objects (air, surface, subsurface contacts). Each track is composed of up to three distinct graphic symbols. The most basic representation of a track is a "dot" representing the geographic position of the track. Additionally, there is a standard (NTDS) set of shapes, used for track categorization/identification, which may be superimposed over the dot (see Figure 21). The third component is the Velocity Leader (see subparagraph (14) for a detailed description).

The system Tracking function shall generate and display all Air, Surface and Subsurface tracks as specified in Figure 21.

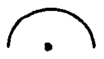
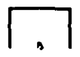
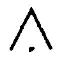
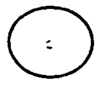
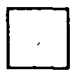
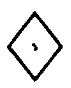
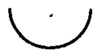
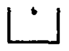
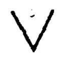
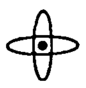
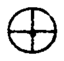
	Friendly	Unknown	Hostile
Air			
Surface			
Subsurface			

Figure 21 : LCCDS Track Symbology

(6) Tentative Track: 

New tracks generated by local sensors (radar) shall appear on the TacPlot as Tentative. Once a valid course and speed is established for the Tentative track the symbology will change to reflect the correct Track category as described in Figure 22.

(7) Ownship position: 

Ownship position symbol shall be located in the center of the TacPlot by default. Ownship position may be slewed to any other point on the TacPlot while maintaining correct spatial relationship with all other TacPlot symbology. There shall be only one Ownship position.

(8) Navigation Hazard: ⤳

The Navigation Hazard symbol shall be available to the user for marking the geographic position of known navigational hazards on the TacPlot. Up to 50 may be displayed at once.

(9) Man in Water: ◉

This symbol shall be available to the user for marking the geographic position of a man in the water. Up to 7 may be displayed on the TacPlot at once. It can be entered onto the TacPlot at an position using the Hook function. If no position is hooked, then the Main in Water symbol is entered at the current Ownship position.

(10) Waypoint: X

Functions and program action associated with this symbol is described in section Goal 4.1.2.1.

(11) Data Link Reference Point: ☒

The DLRP represents a fixed geographic reference position common to all Link 11 participating units. There shall be only one DLRP symbol active in the system at one time.

(12) Formation Center: ┼

The Formation Center is a moving geographic position representing the center of a group of ships steaming in formation. This position shall have the formation course and speed associated with it. There may be up to two Formation Center symbols active in the system at one time.

(13) Position and Intended Movement (PIM): PIM

The PIM function provides time-speed-distance information for progress along a selected steaming route associated with Ownship or Formation Centers.

PIM can be viewed as Ownship or formation planned position based on a pre-computed base course and speed to arrive at destination at the required time. The PIM function associated with Ownship, and/or Formation Center, shall provide the following navigational information based on destination time and place and associated base course and speed:

- display the PIM symbol at the planned position along the steaming route based on current or user selected time.
- PIM shall maintain a running calculation of distance and time ahead/behind plan based on current geographic position.
- provide recommended course and speed for Ownship in order to regain PIM position.

(14) Velocity Leaders:

The Velocity Leader is a vector (line) of variable length and direction associated with each track displayed on the TacPlot. Vector origin is the center (dot) of the track symbol with direction representing track course and length varying proportionally with track speed. Velocity leaders shall be displayed with all active track symbols on the TacPlot unless the user chooses to filter them out. Figure 22 provides a depiction of a Velocity Leader associated with a Track classified as Hostile Air.

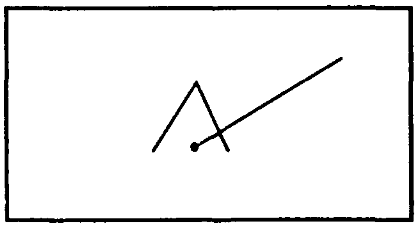


Figure 22 : Velocity Leader

Velocity Leaders shall be displayed by default when course and speed information for any track becomes available. The user shall have the capability to toggle Velocity Leaders on/off for all tracks, specified categories, or individually using the hook function.

The default length value for Velocity Leaders shall be based on 12 minutes of track travel at current track speed. This length value shall be operator variable between 3 and 90 minutes.

G 1.2.1.2: Track Category Select Function

This function shall provide the user with a selection matrix of possible track categories and symbolic representations. The categories are outlined in Figure 21. In addition to providing the category select feature, this matrix also provides selection of various symbolic representations for any category or individual track. The options available follow:

(1) Normal:

The graphic shape representing the track category (see Figure 21) of any or all associated tracks is displayed in normal size as specified by NTDS standards [Ref. 13].

(2) Enlarged:

The graphic shape representing the track category (see Figure 21) of any or all associated tracks is displayed in twice normal (enlarged) size as specified by NTDS standards [Ref. 13].

(3) Suppressed:

The graphic shape representing the track category (see Figure 21) of any or all associated tracks is not displayed. Suppressed tracks are represented by a "dot" only, located at current geographic position.

G 1.2.1.3: TacPlot Display Control Functions

The following display control functions have been adapted for LCCDSWS from existing CDS and C^3 systems [Ref. 7, Ref. 15]

(1) Hook:

The Hook function identifies the position of a symbol or arbitrary point on the TacPlot for use by an active function, or for subsequent use by a function. Only one symbol or point on the TacPlot can be hooked at any time.

(2) Recenter On Ownship:

The Recenter On Ownship function reorients the TacPlot with the Ownship symbol positioned in the center of the window. All displayed Tracks and Special Points shall move to retain correct spatial relationships relative to Ownship and TacPlot center.

(3) Recenter On Lat/Long:

The Recenter On Lat/Long function reorients the TacPlot, centering the window at a desired geographic position. All displayed Tracks and Special

Points shall move to retain correct spatial relationships relative to Ownship and TacPlot center.

(4) Recenter On Hook:

The Recenter On Hook function reorients the TacPlot, centering the window at the current hooked position. All displayed Tracks and Special Points shall move to retain correct spatial relationships relative to Ownship and TacPlot center.

(5) Read Lat/Long:

The Read Lat/Long function displays the geographic latitude and longitude coordinates of any hooked position (symbol or arbitrary point) on the TacPlot.

(6) Display Lat/Long:

The Display Lat/Long function displays a single Track History point at the geographic position associated with a set of latitude and longitude coordinates entered by the user.

(7) Amplify:

The Amplify function shall display alphanumeric amplifying information either selectively for any hooked symbol, or generally for all displayed symbols.

(8) Increase/Decrease Scale:

TacPlot Range Scale shall be provided as a information window within the TACPlot at all times. The unit of measure shall be the nautical mile (NM). The Range Scale shall be an integer and a member of the following set of values:

1, 2, 4, 8, 16, 32, 64, 128, 256, 512, 1024

The scale selected is equal to the radius of the largest enscribable circle contained within the TacPlot. See Figure 23 for a graphic representation using 16 nm as an example.

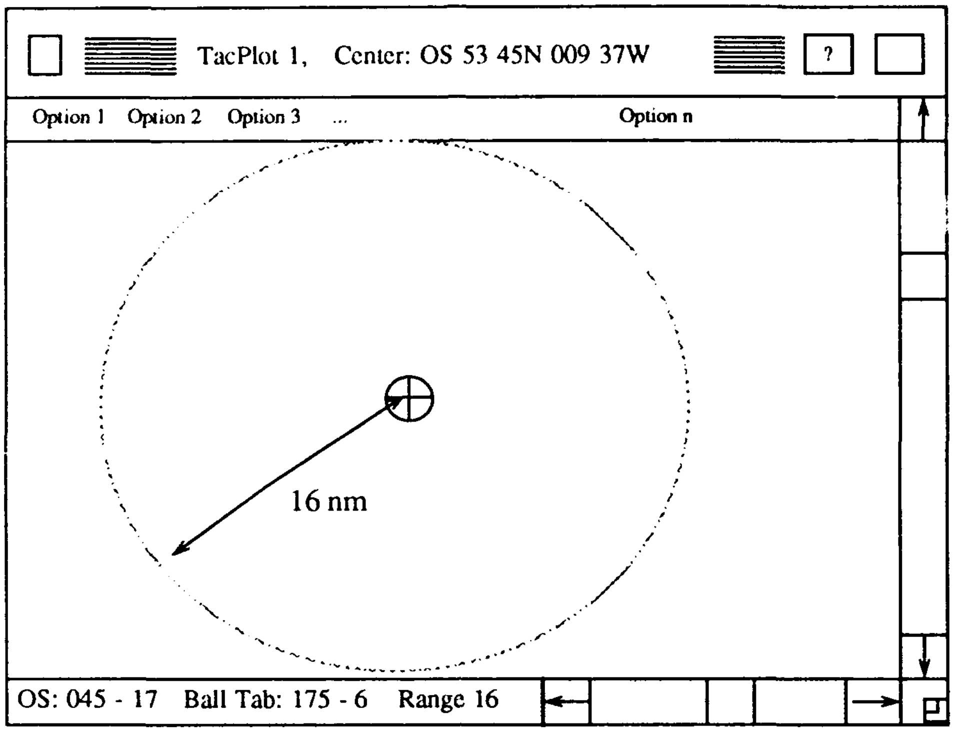


Figure 23 : The Tactical Plot Range Scale

(9) Destroy Symbol:

The Destroy Symbol function removes selected (hooked) symbols from the TacPlot and Tactical Database on an item-by-item basis. Cleared symbols are not removed from the Tactical Database and may be recalled (redisplayed) if desired using the Recall function.

(10) Clear Symbol:

The Clear Symbol function removes selected (hooked) symbols from the TacPlot on an item-by-item basis. Cleared symbols are not removed from the Tactical Database and may be recalled (redisplayed) if desired using the Recall function.

(11) Recall:

The Recall function redisplays previously cleared symbology. The user shall be given a choice of recalling all, or some specified category of, symbols.

(12) Display Link:

This function shall display all Link 11 remote tracks upon user demand.

(13) Assign Track Number:

The system shall automatically assign track numbers to all locally generated tracks based upon a user designated set of integers. Valid track numbers must be four digit integers in the range of 0200-7776 to ensure compatibility with Link 11 remote tracks.

(14) Ownship to Position:

This function provides range and bearing (true or relative) from Ownship to any arbitrary hooked position on the TacPlot. The range and bearing values will update continuously as Ownship moves in relation to the hooked position.

(15) Ownship to Ball Tab:

This function provides range and bearing (true or relative) from Ownship to current Ball Tab position on the TacPlot. The range and bearing values will update continuously as Ownship moves in relation to the Ball Tab position and as the user changes the Ball Tab position. The Range and bearing values shall "float" alongside the Ball Tab symbol as it moves at all times while this function is active.

G 1.2.1.4: TacPlot Graphic Tools

Availability of utility lines, circles and other graphic objects is necessary for the user to manage the tactical picture and enhance tactical decision-making.

Graphics Tools shall be considered a subset of Special Points managed and maintained in the Tactical Database. They shall be defined by lat/long position and dimension attributes. These graphics shall appear as background shapes on the TacPlot and will not obscure Tracks and other Special Points as defined in Goal 1.2.1.1.

Graphic Tools may be filled or unfilled, have variable width borders and have text associated with each object.

As a minimum, available border styles shall support creation of solid, dotted and dashed lines.

The following Graphic Tools shall be available for use on the TacPlot:

(1) Lines:

Lines may originate at any hooked position, symbol or designated lat/long on the TacPlot. The Line function displays a straight line originating from the preceding hooked position to the current position of the Ball Tab, moving with the Ball Tab as the user moves the trackball.

(2) Circle:

Circles may originate at any hooked position, symbol or designated lat/long on the TacPlot. The Circle function displays a circle centered on the preceding hooked position with radius equal to the distance between origin and current position of the Ball Tab, varying with the Ball Tab as the user moves the trackball.

(3) Range Circle:

This function displays a circle whose center is the Ball Tab with fixed, user entered, radius. The range circle will float in conjunction with Ball Tab movement and may be fixed to any desired point on the TacPlot using the Mark function.

(4) Expanding Circle:

This function depicts a circular area of probability of a track/target associated with a time and position on the TacPlot. From origin, a circle expands at a rate determined by a user entered speed estimate for the track.

(5) Moving Circle:

This function generates a circular area of fixed radius which moves across the TacPlot from its origin (hooked position) with a user entered course and speed.

(6) Radar Horizon:

This function depicts the estimated line-of-sight range of the platform's radar based on user entered values of height above waterline, type of radar, operating mode and environmental (weather) effects. While this function is active a circle depicting this radar range estimate shall be displayed with Ownship in its center.

(7) Ellipse:

Ellipses may originate at any hooked position, symbol or designated lat/long on the TacPlot. The Ellipse function displays an ellipse centered on the desired position with user entered values for major and minor axes. This function may also operate in a "rubber band" fashion where the shape is stretched and sized based on Ball Tab movement in relation to a previously hooked position. The Ellipse may be slewed to any position on the TacPlot and rotated up to 180 degrees before being fixed using the Mark function.

(8) Rectangle:

Rectangles may originate at any hooked position, symbol or designated lat/long on the TacPlot. The Rectangle function displays a square or rectangular shape centered on the desired position with user entered length/width values. If only one value is entered, a square will be generated. This function may also operate in a "rubber band" fashion where the shape is stretched and sized based on Ball Tab movement in relation to a previously hooked position. The Rectangle may be slewed to any position on the TacPlot and rotated up to 180 degrees before being fixed using the Mark function.

(9) Arc:

Arcs may originate at any hooked position, symbol or designated lat/long on the TacPlot. This function shall operate in a "rubber band" fashion where the shape is stretched and sized based on Ball Tab movement in relation to a previously hooked position. The Arc may be slewed to any position on the TacPlot and rotated up to 180 degrees before being fixed using the Mark function.

(10) Mark:

The Mark function fixes a previously drawn graphic object to its current position on the TacPlot.

(11) Save:

The Save function operates on user generated graphic objects which exist on the TacPlot allowing them to be saved in the Tactical Database as user defined instantiations of the USERDEFINED (see Goal 3.3) class of database objects.

G 1.2.1.5: Map Display

The LCCDSWS shall provide the capability to superimpose shoreline maps on the selected TacPlot display.

G 1.2.1.5.1: Map Database

The user shall have access to available World Vector Shoreline (WVS) or DMA Digital Terrain Elevation (DTED) map data stored in a separate database.

G 1.2.1.5.2: Map Functions

G 1.2.1.5.2.1: Map Display Function

The default map displayed shall be that part of the shoreline visible based on current TacPlot range scale and geographic position (lat/long) of the TacPlot display center. Appropriate user cues and alerts shall be generated, via Function I/O windows, when the Map Display function is initiated and the nearest shoreline is either off scale or completely out of range.

G 1.2.1.5.2.2: Map Select Function

Any section of shoreline may be arbitrarily selected for display by entering a lat/long into an appropriate Function I/O window. Any shoreline within maximum displayable range centered around this point shall be displayed.

Pre-defined shoreline segments, ports, harbors, chokepoints etc, shall be user selectable, via menu choice, for display on a specified TacPlot. Additionally, the user shall be capable of defining and storing associated navigational, tactical and other information associated with these pre-defined maps.

G 1.2.1.5.2.3: Bottom Contours

Bottom contour lines necessary for coastal navigation and operations shall be available for display in conjunction with shoreline maps.

G 1.2.1.5.2.4: Display Grid Lines

Latitude and longitude grid lines are user selectable anytime regardless of proximity of landmasses or displayed maps. They shall be toggled on/off, displayed at either full or half degree increments, and be represented by solid, dotted or dashed lines as selected by the user.

G 1.2.1.5.2.4.1: Coordinate Labels

Latitude and longitude values associated with each grid line shall be selectable (on/off) by the user. When selected on, latitude labels shall appear on either side, longitude labels across the top and bottom of the TacPlot, superimposed over the appropriate grid line.

G 1.2.1.5.2.5: Landmass Fill

The landmass side of any displayed shoreline shall be fillable using color or fill patterns as desired.

G 1.2.1.5.2.6: Refresh

The displayed segment of a shoreline map shall be automatically updated (refreshed) as the TacPlot tactical picture changes. Map refresh shall be necessary for the following conditions:

- user demand (via a Map Refresh function)
- Ownship position change
- Change of TacPlot range scale
- Use of any of the TacPlot Recentering functions

G 1.2.1.5.2.6.1: Map Refresh

This function is available from the Map menu as a single action switch which updates the currently displayed shoreline map segment or redisplays a previously cleared shoreline map.

G 1.2.1.5.2.7: Map Clear

This function allows the user to suspend display of the map. The cleared map may be updated and redisplayed by use of the Map Refresh function.

G 1.2.1.6: Sensor Selection

The TacPlot shall maintain an indication of the sensor selected for each TacPlot Window as applicable. An example for a sensor selection window is provided by Goal 1.1.1.3.3 and shall be available for each TacPlot window.

G 1.3: Display Doctrine

The system shall provide, via a *Display Doctrine*, the capability for user defined conditional statements, in IF-THEN form, which control data filtering and specify presentation parameters for displayed data. These conditional statements are the building blocks for the Display Doctrine.

G 1.3.1: Display Doctrine Scope

There shall be one Display Doctrine associated with each instantiation of the Graphics window class. In other words, each active TacPlot has its own independent Display Doctrine. The Display Doctrine for one TacPlot instantiation shall have the capability to be copied directly into the Display Doctrine of any other active TacPlot windows.

The user shall have the capability to define a set of conditions which, if met, will cause some display function, or set of functions, to effect the Tactical Display in some manner to tailor the displayed tactical information as necessary for the mission and to alert the user when any specified event or situation occurs.

G 1.3.2: Display Doctrine Structure

The Display Doctrine shall be comprised of a set of one or more IF-THEN conditional statements. Each Display Doctrine statement shall consist of two clauses; an IF clause specifying certain conditions to be met, and an associated THEN clause specifying the appropriate action to be taken. Figure 24 depicts the general syntax for a single Display Doctrine IF-THEN statement..

These statements shall be simple and easily defined using window-based templates and forms with appropriate information fields. These fields shall have appropriate choices tied to pop-up menus, tableaus and lists as necessary to aid user selection.

The choice of appropriate IF conditions and associated THEN operations on the data presentation shall be context sensitive. In other words, the menu choices offered must correspond to the data attributes of the object (i.e. a TacPlot window, track, or category of tracks) which the Display Doctrine statement must operate upon.

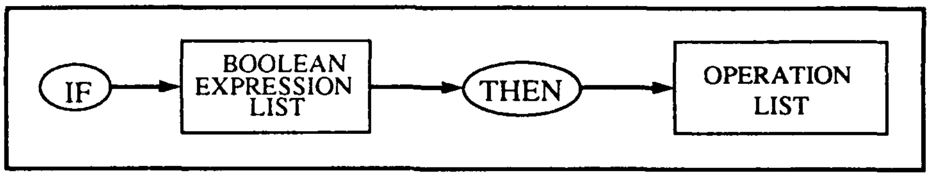


Figure 24 : Display Doctrine Syntactic Structure.

G 1.3.2.1: The IF Clause

The IF clause shall specify the condition, or set of conditions, to be met. The IF clause of any statement shall allow up to at least 12 qualifying conditions, connected together using either the AND or the OR logical operators.

G 1.3.2.1.1: Logical Connectives

The logical connectives AND and OR shall be used to link the qualifying conditions (if more than one) of the IF clause. For the purposes of clarity and simplicity of each individual IF-THEN statement, the AND and OR connectives shall be considered *mutually exclusive logical operators.* When the user indicates more than one qualifying condition in the IF clause he must choose one or the other. *This operator shall connect all qualifying conditions in the IF clause for that statement.*

G 1.3.2.1.2: IF Clause Qualifying Conditions

The qualifying conditions shall be specified using boolean expressions. These expressions are defined as comparisons between a specified attribute of some object and a threshold value associated with that attribute.

The result of any comparison shall assign a boolean value (TRUE/FALSE) to the associated qualifying condition. The logical connectives AND and OR make the overall determination of TRUE or FALSE for the IF clause as follows:

(1) AND Operator. If any (one or more) qualifying conditions in an IF clause resolves to FALSE then the IF clause as a whole is FALSE and the corresponding THEN clause is not executed. All qualifying conditions comprising the IF clause must resolve to TRUE in order for the THEN clause to be invoked.

(2) OR Operator. If any qualifying condition in an IF clause, containing one or more qualifying conditions, resolves to TRUE then the IF clause as a whole is TRUE and the corresponding THEN clause is executed. Any one qualifying condition comprising the IF clause must resolve to true in order for the THEN clause to be invoked.

The following boolean operators (comparators) shall be available for use in defining the comparisons:

- equal (=)
- not equal (<>)
- less than (<)
- greater than (>)
- less than or equal (<=)
- greater than or equal (>=).

G 1.3.2.1.3: Boolean Expression List

The IF clause shall consist of a boolean expression list containing one or more expressions linked using one of the logical connectives. The syntax required for construction of any boolean expression is outlined in Figure 25. The system shall provide the user with a "fill-in-the-blank" style template supporting at least 12 expressions. Along with the templates shall be associated support windows which provide the user with an appropriate range of choices for each information input field. Input fields shall be filled by the user via alphanumeric input and/or "checkbox" style menu selections. Each information field must be context sensitive in that it has the capability of alerting the user when an invalid or inappropriate entry has been made. Additionally, the system shall evaluate each completed expression, as a whole, to ensure appropriate semantic content (to protect against syntactically correct nonsensical operations).

G 1.3.2.1.4: Expression Attributes and Threshold Values

Each boolean expression shall evaluate to TRUE or FALSE based on the logical comparison of the current value of a specified Attribute and the specified Threshold Value for that Attribute. These Attributes appropriate for selection, and range of values associated with each, shall be provided to the user to assist construction of semantically correct Display Doctrine function definitions.

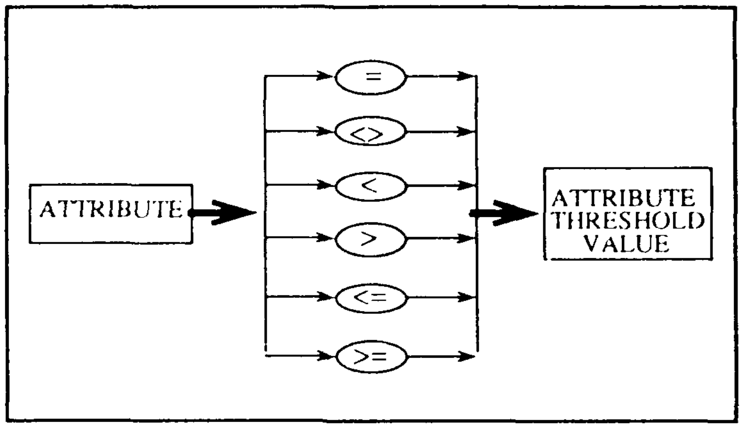


Figure 25 : Boolean Expression Structure

G 1.3.2.2: The THEN Clause

The THEN clause shall specify the operation to be invoked when (if) the condition, or set of conditions, is met.

These operations should include, but not be limited to, the following types of display related activity: blinking, enlarging or otherwise enhancing tactical symbology, displaying pop-up cues, messages, alerts, tableaus or on-line checklists.

G 1.3.2.2.1: THEN Operations Definition

The THEN clause specifies the operations which will act on the TacPlot graphics display whenever the associated IF conditions are fulfilled. There shall be two types of operation, categorized as *default* and *user-defined*, available for use by any Display Doctrine statement.

G 1.3.2.2.1.1: Default Operations

There are certain display characteristics which are part of the basic CDS and already available through the Track Category Select function (Goal 1.2.1.3). These characteristics, therefore by default, are available for use by the Display Doctrine IF-THEN statements.

The following default operations shall be available for use as Display Doctrine operations:

- Normal
- Enlarged
- Suppressed
- Toggle Velocity Leader On/Off
- Extend Velocity Leader

G 1.3.2.2.1.2: User-Defined Operations

In addition to the Default operations available for use are any pre-defined operations specified by the user. Any appropriate user-defined operations may be added to the current menu/list of available THEN operations for any given Display Doctrine statement.

User-defined operations shall be constructed separately from, but in a similar manner as, the Display Doctrine statements. The system shall provide the user with "fill-in-the-blank" style templates to quickly and easily guide the user through the necessary choices for building various sorts of display operations. Along with the templates shall be associated support windows which provide the user with an appropriate range of choices for each information input field. Input fields shall be filled by the user via alphanumeric input and/or "checkbox" style menu selections. Each information field must be context sensitive in that it has the capability of alerting the user when an invalid or inappropriate entry has been made. See Figure 26 for a graphic example.

The following list provides some examples of user-defined operations which should be available for use as Display Doctrine operations:

- blinking symbology.
- opening user defined alphanumeric or function windows with cues, alerts and information tailored to specified tactical conditions and situations.
- changing or intensifying colors, using inverse video.
- displaying user defined graphic objects or icons.

G 1.3.3: Display Doctrine Modification

The user shall have the capability to modify any Display Doctrine statement in order to change either the Boolean Expression List or the

Operation List. Statements may be added or deleted to any Display Doctrine as desired by the user.

G 1.3.4: Display Doctrine Consistency Checking

The system shall ensure that all statements entered into the Display Doctrine are consistent with statements already existing. The Display Doctrine associated with each instantiation of a Graphics Window (TacPlot) can be viewed as a set of user requirements for that window. The system shall be capable of determining that a newly entered statement conflicts with some previous one. In such cases the system shall alert the user. *Under no circumstance shall the system allow conflicted statements to go unresolved.*

G 1.3.5: Display Doctrine Applications

The Display Doctrine shall, as a minimum, apply to and operate on the following types of displayed information:

- all Tracks qualifying for display
- all tracks of a particular Track Category qualifying for display
- any single Track or Special Point identified using the Hook function.

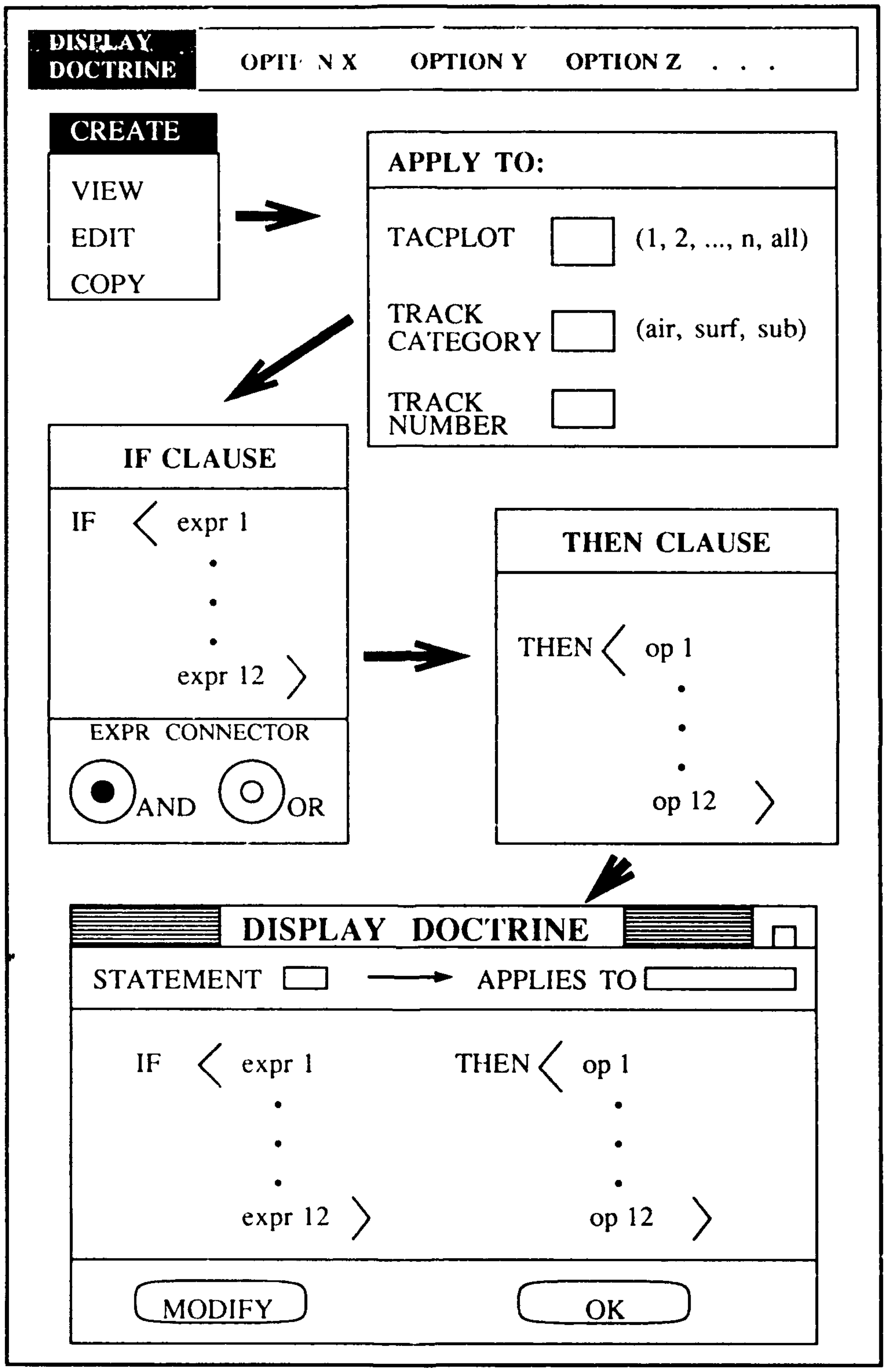


Figure 26 : Display Doctrine Statement Construction

2. System Control Function

a. Preface

The system control function is neither requested by the customer nor described in one of the increments. However, the study of an existing CDS [Ref. 7] made it clear, that a collection of functions to control, monitor and initialize the system will be necessary.

b. Goals

G 2: System Control Function

The system has to control, initialize and monitor the LCCDS hardware and software configurations. We refer to this function as the *System Control Function.*

The System Control Function shall be supported by a set of menu choices, dialogs, cues and tableaus, which allow the user to initialize, manage the I/O configuration with external system and select display characteristics such as color, window positions.

G 2.1: Initialization:

Upon initialization the system shall require entry of the following navigational parameters :

G 2.1.1: Ownship Position

Ownship position (automatic entry from Navigational System when enabled, otherwise manual).

G 2.1.2: Greenwich Mean Time

Greenwich Mean Time (automatic entry from Navigational System when enabled, otherwise manual).

G 2.1.3: CPA Parameters

CPA Parameters (manually):

- Close CPA Range: 200 - 9,999 yards,

- Close CPA Time : 5 - 99 minutes,
- Collision CPA Range: 1 - 199 yards,
- Collision CPA Time: 5 - 99 minutes.

The system shall allow the user to save the CPA parameters for future use.

G 2.1.4: Data Link Reference Point (DLRP)

Enter the DLRP (manually); if no DLRP is entered it shall be set to ownship initial position, thereafter DLRP shall be updated by manual entries.

G 2.1.5: Julian Date

Enter the Julian Date (manually); it will be used for user information only.

G 2.1.6: Initial Values

If the navigational system is ready and operational upon initialization the I/O channel shall be enabled by preset, otherwise disabled.

In case the Navigational System is enabled, in addition to ownship position and GMT, ownship course and speed values shall be accepted from the Navigational System. Otherwise course and speed shall be set to 0, thereafter course and speed shall be updated by manual and automatic updates.

If the navigational parameters are not entered within 5 minutes of initialization, the system shall notify the user.

G 2.2: Configuration Control

The system shall provide the user with a choice of available input/output channels for external systems (i.e. Radar, Link 11, Navigation).

G 2.2.1: I/O Channels

The system shall provide the capability for identifying up to 20 separate I/O channels. The user will have control over the channel configuration via the MMI. An example for such a control mechanism (Interface Control Window) is described in Goal 1.1.1.3.4.

G 2.3: Configuration Monitoring

The system shall provide the capability to monitor hardware and software configurations for both LCCDS and external systems.

G 2.3.1: LCCDS System Status.

The system shall provide for error detection and notification. Changes to the system status not initiated by the user shall result in display of an appropriate warning window and automatic update of the System Status Tableau (Goal 4.6.2.10).

G 2.3.2: Software Monitoring

LCCDSWS shall provide the user with graphic feedback, in the form of warning windows, cues and alerts, for any user initiated operations which are not within the limits of established functionality, or cannot be complied with based on current configuration or in response to an actual system software fault.

G 2.3.3: Recovery Data

The system shall provide the user with the capability to specify some subset of system data for system recovery in the event of equipment failure, error conditions or a program fault. This is accomplished by specifying the data and the periodicity for which that data shall be saved to some secondary storage device.

G 2.4: Training Function

The system has to support the determination of the LCCDS operational readiness through the conduct of training exercises. We refer to this function as the *Training Function.*

Requirements for the prototype system will not cover the refinement for the Training Function. However, it should be recognized that embedded training functions, developed concurrently with the actual system, will have a large impact on fleet acceptance and operator training at the unit and individual platform level.

G 2.5: Test Function

The system has to support the determination of the LCCDS material readiness through the conduct of tests and associated fault processing. We refer to this function as the *Test Function*.

Requirements for the prototype system will not cover the refinement for the Test Function.

3. The Tactical Database Function

a. Preface

The Tactical Database is one of the important integral parts of the LCCDSWS. NAVSEA requires [Appendix A] the design/development or selection of an object-oriented database management system (OODBMS). Ross [Ref. 16] evaluates three commercially available OODBMS as candidates for the LCCDS project and provides a survey of object-oriented concepts. To identify the goals for the Tactical Database we suggest to apply an object-oriented development method as described in Booch [Ref. 17]. The design of a system is performed by the following steps [Ref. 17: p. 48]:

1. Identify the objects and their attributes.
2. Identify the operations that affect each object and the operations each object must initiate.
3. Establish the visibility of each object in relation to other objects.
4. Establish the interface of each object.
5. Implement each object.

However, steps 3, 4 and 5 are dependent on or provided by the OODBMS system to be chosen. These steps shall be the subject of further research work when an OODBMS system is selected and available.

Goals 3.2 and 3.3 try to identify objects for the Tactical Database, their attributes and operations. It is assumed that these sets of objects, attributes and operations represent minimum sets and may require further refinement.

Goal 3.4 identifies basic input and output operations as they apply to all objects in the data base or on particular objects with respect to some attributes (e.g., objects with origin "Local Auto" perform I/O operations with the Radar Interface). Again, it is necessary to further refine these operations since the interfaces are not yet defined.

b. Goals

G 3: Tactical Database

The system shall provide a Tactical Database. We refer to this function as the *Tactical Database Function.*

G 3.1: Integrate OODBMS into LCCDSWS

The Tactical Database shall be an object oriented database management system (OODBMS). It shall be an integral part of the LCCDSWS.

G 3.2: Provide a Set of Object Types

The OODBMS contains information necessary to establish an accurate picture of a ship's environment. The unit of such information will be an object within the OODBMS. Information necessary to generate an object can be provided by the user, an external system (Radar, Link 11) or an internal module. The OODBMS shall provide features for pre-defined objects (statically instantiated objects) and for objects to be determined at runtime (dynamically instantiated objects). Objects are described by their attributes and operations.

G 3.2.1: Types of Instantiated Objects

The following identifies types (classes) of statically instantiated objects, a a minimum set of their attributes and operations in terms of basic CDS functions as described in Goal 3.

Class: OWNSHIP

Attributes: Geographical Position {latitude and longitude}

Time of Position {hh:mm:ss}

Course {degrees}

Speed {knots}

Origin {Local Manual}

Tracknumber {positive integer 0 .. 9999}

Type {String of 10 characters}

Track History {<boolean>}

Operations: G 3.1.1: Monitor Ownship Position,

G 4.1.1.2: Navigational Computation,

G 4.1.2.2: CPA Processing.

Symbology: An object of class OWNSHIP is associated to the ownship symbol as described in Goal 1.1.1.1 (7).

Class: TENTATIVE TRACK

Attributes: Geographical Position {latitude and longitude}

Relative Position {bearing in degrees and distance in nm}

Time of Position {hh:mm:ss}

Course {degrees}

Speed {knots}

Category {TENTATIVE}

Identity {UNKNOWN}

Origin {Local Manual, Local Auto}

Tracknumber {positive integer 0 .. 9999}

Type {String of 10 characters}

Operations: G 4.2.1: New Track Establishment,

G 4.2.2: Track Position Data,

G 4.2.3: Track Course and Speed Determination,

G 4.2.4: Dead Reckoning,

G 4.3.1.1: Initial Category Assignment,

G 4.3.2.1: Initial Identity Assignment,

G 4.2.7.1: Track Termination.

Symbology: Objects of class TENTATIVE TRACK are associated to the tentative track symbol as described in G 1.1.1.1: (6)

Class: AIR TRACK

Attributes:	Geographical Position {latitude and longitude}
	Relative Position {bearing in degrees and distance in nm}
	Time of Position {hh:mm:ss}
	Course {degrees}
	Speed {knots}
	Height {feet}
	Category {AIR}
	Identity {UNKNOWN or FRIENDLY or HOSTILE}
	Origin {Local Manual, Local Auto, Remote}
	Tracknumber {positive integer 0 .. 9999}
	Type {String of 10 characters}

Class: SURFACE TRACK

Attributes:	Geographical Position {latitude and longitude}
	Relative Position {bearing in degrees and distance in nm}
	Time of Position {hh:mm:ss}
	Course {degrees}
	Speed {knots}
	Category {SURFACE}

Identity {UNKNOWN or FRIENDLY or HOSTILE}

Origin {Local Manual, Local Auto, Remote}

Tracknumber {positive integer 0 .. 9999}

Type {String of 10 characters}

Class: SUBSURFACE TRACK

Attributes: Geographical Position {latitude and longitude}

Relative Position {bearing in degrees and distance in nm}

Time of Position {hh:mm:ss}

Course {degrees}

Speed {knots}

Depth {feet}

Category {SUBSURFACE}

Identity {UNKNOWN or FRIENDLY or HOSTILE}

Origin {Local Manual, Local Auto, Remote}

Tracknumber {positive integer 0 .. 9999}

Type {String of 10 characters}

Operations : for classes AIR, SURFACE and SUBSURFACE

G 4.1.2.2: CPA Processing,

G 4.2.2: Track Position Update,

G 4.2.3: Track Course and Speed Determination,

G 4.2.4: Dead Reckoning.

G 4.2.5: Track Position Prediction,

G 4.2.6: Track History Processing,

G 4.2.7.2: Manual Termination

G 4.3: Identification Function..

Note: Due to several similarities in the attribute sets of the above classes it would probably be appropriate to make these classes subclasses of a class TRACK.

During further refinement and implementation it should be considered that the set of operations for these objects depends on the attribute Origin (Local Manual -> manual tracking functions; Local Auto -> auto tracking functions, Remote -> no tracking).

Symbology: Objects of class AIR TRACK, SURFACE TRACK and SUBSURFACE TRACK are associated with the track symbols as described in Goal 1.1.1.1(5), Figure 21.

Class: REFERENCE POINT

Attributes: Geographical Position {latitude and longitude}

Relative Position {bearing in degrees and distance in nm}

Time of Position {hh:mm:ss}

Category {SPECIAL}

Identity {REFPOINT}

Origin {Local Manual}

Operations : G 4.1.3.1.3: Enter /update Reference Point,

G 4.1.2.2: CPA Processing,

Symbology: Objects of class REFERENCE POINT are associated with the Reference Point symbol as described in Goal 1.1.1.1(4).

Class: NAVIGATION HAZARD

Attributes: Geographical Position {latitude and longitude}

Relative Position {bearing in degrees and distance in nm}

Time of Position {hh:mm:ss}

Category {SPECIAL}

Identity {NAVHAZ}

Origin {Local Manual}

Operations : G 4.1.3.1.3: Enter /update Navigation Hazard,

G 4.1.2.2: CPA Processing,

Symbology: Objects of class NAVIGATION HAZARD are associated with the Navigation Hazard symbol as described in Goal 1.1.1.1(8).

Class: MAN IN WATER

Attributes: Geographical Position {latitude and longitude}

Relative Position {bearing in degrees and distance in nm}

Time of Position {hh:mm:ss}

Category {SPECIAL}

Identity {MAN IN WATER}

Origin {Local Manual}

Operations : G 4.1.3.1.1:Enter /update Man in Water,

G 4.1.2.2: CPA Processing,

Symbology: Objects of class MAN IN WATER are associated with the Man in Water symbol as described in Goal 1.1.1.1(9).

Class: WAYPOINT

Attributes: Geographical Position {latitude and longitude}

Relative Position {bearing in degrees and distance in nm}

Time of Position {hh:mm:ss}

Steaming Route {integer 1..6}

Category {SPECIAL}

Identity {WAYPOINT}

Origin {Local Manual}

Operations: G 4.1.2.1: Waypoint Geometry.

Symbology: Objects of class WAYPOINT are associated with the Waypoint symbol as described in Goal 1.1.1.1(10).

Class: DATA LINK REFERENCE POINT

Attributes: Geographical Position {latitude and longitude}

Time of Position {hh:mm:ss}

Category {SPECIAL}

Identity {DLRP}

Origin {Local Manual}

Operations : G 4.1.3.1.1:Enter /update DLRP,

G 4.1.2.2: CPA Processing,

Symbology: Objects of class DATA LINK REFERENCE POINT are associated with the Data Link Reference Point symbol as described in Goal 1.1.1.1(11).

Class: FORMATION CENTER

Attributes: Geographical Position {latitude and longitude}

Time of Position {hh:mm:ss}

Relative Position {bearing in degrees and distance in nm}

Course {degrees}

Speed {knots}

Category {SPECIAL}

Identity {FC}

Origin {Local Manual}

Operations : G 4.1.3.1.2:Enter /update Formation Center,

G 4.2.4: Dead Reckoning,

G 4.1.2.2: CPA Processing,

Symbology: Objects of class FORMATION CENTER are associated with the Formation Center symbol as described in Goal 1.1.1.1(12).

Class: POSITION AND INTENDED MOVEMENT

Attributes: Geographical Position {latitude and longitude}

Relative Position {bearing in degrees and distance in nm}

Time of Position {hh:mm:ss}

Course {degrees}

Speed {knots}

Category {SPECIAL}

Identity {PIM}

Origin {Local Manual}

Operations : G 4.1.3.1.2:Enter /update PIM,

G 4.2.4: Dead Reckoning,

G 4.1.2.2: CPA Processing,

Symbology: Objects of class POSITION AND INTENDED MOVEMENT are associated with the Position and Intended Movement symbol as described in Goal 1.1.1.1(12).

Note: Due to several similarities in the attribute sets of the above classes it would be appropriate to make these classes subclasses of a class SPECIAL POINT.

G 3.3: Definition of New Classes

The OODBMS shall allow the user to define new classes of objects at runtime by choosing from a set of attributes and operations. The definition of a new class shall allow the user to instantiate objects of that class.

For the definition of a new class the user shall be allowed to construct the definition by selection from a given set of attributes and operations or by defining new attributes or operations.

It is recommended that attributes be further divided in specific and generic attributes. Specific attributes may include:

- position {lat/long}
- course {degrees}
- speed {knots}
- relative position {bearing/distance}
- category {AIR, SURFACE, SUBSURFACE}.

Generic attributes may include:

-character/textfields

-numeric fields

-floating value fields

-date fields

-time fields

-logical fields.

In the following we provide an example for a user defined class:

Class: USERDEFINED

- **Attributes:** minimal set is:

  Geographical Position {latitude and longitude}

  Time of Position {hh:mm:ss}

  C character/textfields

  N numeric

  F floating value

  D date

  T time

L Logical

DD <condition>

Symbology: Objects of class USERDEFINED are associated with the Userdefined symbol as described in Goal 1.2.1.4.

G 3.4: Provide Input and Output for OODBMS

The OODBMS shall provide features for data entries and output.

G 3.4.1: Interaction with MMI

The OODBMS shall interact with the MMI.

G 3.4.1.1: MMI Input

The OODBMS shall accept all user input operations through the MMI. Input shall be graphical input, alphanumerical input or functional input.

G 3.4.1.1.1: Graphical Input

A graphical input is an input from the graphic I/O portion of the MMI as a result of a user input on the TacPlot converted to a geographical position.

The OODBMS shall accept the geographical position and perform one of the following depending on user specified functions/conditions:

- "hooking" of an object that can be associated with a symbol located at that position.
- instantiation of an object of class: TENTATIVE TRACK, "hooking" of that object and provide for operations as described in Goal 3.2.1.
- instantiation of an object of class: SPECIAL POINT, "hooking" of that object and provide for operations as described in Goal 3.2.1.

G 3.4.1.1.2: Alphanumerical Input

An alphanumerical input is an input from the alphanumeric I/O portion of the MMI as the result of a user input to an alphanumeric I/O window The OODBMS shall accept the alphanumerical input and update the objects that can be associated with that input (e.g., objects being hooked, conditions specified by user for a set of objects).

G 3.4.1.1.3: Functional Input

A functional input is an input from the Function I/O portion of the MMI as the result of a user manipulation to a functional window (button, icon, etc.). The OODBMS shall accept the functional input and perform the operations on applicable objects as described in Goal 4.

G 3.4.1.2: MMI Output

The OODBMS shall output all data to the user through the MMI.

G 3.4.1.2.1: Graphical and Alphanumerical Output

The OODBMS shall automatically generate a stream of objects for use by the MMI Graphic I/O and Alphanumeric I/O:

(1) for graphical output; all objects that qualify for output.

(2) for alphanumeric output:

- the object of class OWNSHIP at all time.
- one hooked object that qualifies for output.
- by user functions specified objects.

The change of an alphanumeric output shall result in an alphanumeric input operation.

G 3.4.2: Interaction with Tracking System

The OODBMS shall interact with a Tracking System.

G 3.4.3: Interaction with Radar Interface

The OODBMS shall interact with the Radar Interface.

G 3.4.4: Interaction with Link 11 Interface

The OODBMS shall interact with the Link 11 Interface.

G 3.4.5: OODBMS System Alerts

The OODBMS shall provide system alerts for output on the alphanumeric I/O portion of the MMI when:

- an invalid operation on an object is issued;
- an invalid condition for an object is specified;
- a relationship between objects cannot be established.

G 3.5: Query Language

The goal for the query language can not be further refined at this time. Specifics of the query language will depend on the OODBMS to be selected or to be designed.

4. The Basic CDS Functions

a. Preface

The functions as described in the following are mainly extracted from Reference 7. They represent the subset of CDS functions applicable to LCCDS. In the context of LCCDSWS these functions can be classified as follows:

(1) CDS functions that are operations on objects as specified in G2; these operations fall in two categories:

- Manually functions that are initiated on user request at the MMI; e.g. a manual position update
- Automatic functions that are performed by the system without user interference.

(2) CDS functions that represent new classes not covered by Goal 3.

(3) CDS functions that represent modules within the LCCDSWS.

b. Goals

G 4: The Basic CDS Functions

The system has to support the user to establish an accurate picture of a ships environment. We refer to this function as the *Basic CDS Function*.

G 4.1: Ownship Monitor Function.

G 4.1.1: Monitor Ownship Position

The system has to maintain the ownship symbol in the center of the TacPlot as a default.

The system shall maintain current and actual information concerning position and velocity of ownship. It shall accept both manual and automatic navigational inputs.

G 4.1.1.1: Manual Inputs

The system shall provide the operator with the capability to enter the following navigation inputs:

- Enable/Disable I/O channel to Navigational System;
- Julian Date;
- Marktime (Nav Sys must be enabled);
- Greenwich Mean Time (GMT) (Nav Sys must be enabled);
- Ownship heading (Nav Sys must be enabled);
- Ownship speed (Nav Sys must be enabled);
- Ownship latitude and longitude (Nav Sys must be enabled);
- True or relative bearing and range to a Special Point.

G 4.1.1.2: Automatic Inputs

The system shall accept the following navigational inputs from a Navigational System:

- Ownship latitude and longitude
- Ownship heading
- Ownship speed
- GMT

G 4.1.1.3: Navigational Computation

The system shall update ownship position and velocity at a minimum of once every four seconds based on the positional data received from either manual or automatic input.

The system shall provide the following capabilities for manually repositioning ownship:

The system shall provide the capability to reposition ownship relative to a geographically fixed Special Point resulting in adjustments to ownship latitude and longitude;

The system shall provide automatic dead reckoning calculations for ownship latitude and longitude position based on the last position, course and speed.

G 4.1.2: Monitor Ownship Surface Maneuvering

The system shall provide the following ownship maneuvering capabilities:

G 4.1.2.1: Waypoint Geometry

The system has to allow for specification of up to six steaming routes with up to 50 waypoints (destinations) per route.

The system shall determine the course at a specified speed, that a vehicular track or the ownship must take in order to intercept a designated waypoint.

The system shall update the maneuvering geometries every 4 seconds.

Waypoint maneuvering data shall be presented with course, speed, bearing and distance to the waypoint and shall include the GMT of the estimated arrival time at the waypoint, the waypoint designation and Time-To-Go to the waypoint.

G 4.1.2.2: Closest Point of Approach Processing

The system has to provide Closest Point of Approach (CPA) geometry from ownship to any track and between any two tracks upon user request.

When the CPA of a specified track meets specified time and range criteria, the user shall be notified of a Close CPA or Collision CPA as appropriate. The system shall allow the user to manually adjust the criteria for Close CPA and Collision CPA calculations and alert as follows:

- Close CPA Range: 200 - 9,999 yards;
- Close CPA Time: 5 - 99 minutes;
- Collision CPA Range: 1 - 199 yards
- Collision CPA Time: 5 - 99 minutes;

Preset parameters for CPA calculations shall be requested from the user upon system initialization.

G 4.1.3: Monitor Surface Area Management

G 4.1.3.1: Special Points

The system shall provide for the following types of Special Points:

- Man in Water;
- Formation Center;
- Position and Intended Movement (PIM)
- Navigation Hazard;
- Data Link Reference Point (DLRP);
- Reference Point.

G 4.1.3.1.1: Man in Water

The system shall provide the capability to enter and reposition up to ten Man-in-Water special points. The coordinates of entry shall be user specified by ball tab coordinates The position of the Man-In-Water shall be geographically fixed. The Man-in-Water Special Point shall be terminated manually on user request.

G 4.1.3.1.2: Formation Center and PIM

The system shall provide the user with the capability to enter/update the following special points:

- Formation Center (FC)
- Position and Intended Movement (PIM)

Entries shall include latitude, longitude, course and speed of the designated special points.

The user shall have the capability to enter, slave or reposition the special point using ball tab coordinates. Repositioning shall not cause automatic course and speed recalculation.

Both Special Points shall be terminated upon user request.

G 4.1.3.1.3: Navigation Hazards and Reference Points

The system shall provide the user with the capability to enter/update the special points:

- Navigation Hazard;
- Reference Point.

The entries shall include latitude and longitude of the designated special point. Both Special Points shall be terminated upon user request.

G 4.1.3.1.4: DLRP

The system shall provide the user with the capability to enter/update a DLRP. Entries shall include latitude and longitude of the DLRP. Once entered a DLRP cannot be terminated.

G 4.1.3.2: Chart Processing

The system shall provide the ability to overlay the TacPlot with World Vector Shoreline maps as available from the Defense Mapping Agency or other sources.

G 4.2: Tracking Function.

The Tracking Function is the manual entry and manual or automatic update of position, course and speed of applicable vehicular tracks as required by Increment Two. It represents operations applicable to objects as described in Goal 2.

G 4.2.1: New Track Establishment

The system shall provide the capability to enter new tracks manually.

A newly entered vehicular track shall be initiated as a tentative track.

The system shall determine when local track entries are sufficiently reliable to warrant a firm track status. Firm track status shall be based on the number of updates. Firm/tentative status shall only apply to vehicular tracks.

The tentative tracks shall become firm tracks as a result of one of the following conditions:

- After three manual position updates as described in Goal 4.2.2.1.
- Manual category entered before three position updates in accordance with Goal 4.3.1.2.
- Manual declared firm.

G 4.2.2: Track Position Data

G 4.2.2.1: Track Position Update

The user shall have the capability to perform manual position (range and bearing) update entries for all vehicular tracks.

The result of these position corrections shall be the update of the tracks position in all cases. It shall also result in automatic course and speed computation.

G 4.2.2.2: Track Reposition

The user shall have the capability to reposition vehicular tracks.

The result of the reposition action shall be update of the track's position in all cases. *It shall not cause* automatic course and speed recomputation.

G 4.2.3: Track Course and Speed Determination

The system shall provide for the determination and entry of a track course and speed.

G 4.2.3.1: Course and Speed Computations

The system shall automatically compute the course and speed for the following:

- A newly entered track.
- Any track on which a position update is taken.

G 4.2.3.2: Manual Course and Speed Entries

The system shall provide the capability to enter course and speed manually for the following tracks:

(1) Vehicular Tracks. Course and speed shall be accepted for all vehicular tracks with bearing values in degrees and speed values in up to 9999 knots.

(2) Special Points. Course and speed shall be accepted for the following unslaved unique special Points:

- Formation Center.
- PIM

G 4.2.4: Dead Reckoning

The system shall automatically perform Dead Reckoning computations for all vehicular tracks via an extrapolation of the last known position, course and speed. If no Track Position Update (Goal 4.2.2.1) is received for a vehicular track within 4 seconds, the track shall be automatically repositioned. This shall not cause automatic course and speed computation.

G 4.2.5: Track Position Prediction:

The system shall compute predicted track positions on user selected vehicular tracks via an extrapolation of the last known position, course and speed.

The prediction rate shall be 4 seconds.

G 4.2.6: Track History Processing

The system shall maintain history positional data on all air, surface and subsurface vehicular tracks. The points shall be displayed upon request for an individual track.

Normal manual track history shall be provided for the following:

- Ownship
- Air Vehicular Track
- Surface Vehicular Track
- Subsurface Vehicular Track

G 4.2.7: Track Termination

The system shall provide for the manual and automatic termination of all local tracks by responding to Drop Track notifications by deleting the specified track from the track stores, erasing its symbol on the screen and notifying the user.

G 4.2.7.1: Automatic Termination

The system shall provide for automatic termination of vehicular tracks as follows:

- Tentative Track: if no update is received within 6 minutes for an air track or 12 minutes for an surface track.

Track termination shall be performed on user confirmation only.

G 4.2.7.2: Manual Termination

The system shall provide the capability to manually terminate a specified track.

Tracks that have other tracks slaved to them shall not be dropped until all slaved tracks have been dropped.

Special points that shall not be dropped once they have entered are:

- Ownship
- Data Link Reference Point (DLRP)

G 4.3: Identification Function

The system shall perform the determination of Category and Identity of applicable tracks.

G 4.3.1: Track Category Determination

The system shall perform vehicular track category determination. The categories to be assigned are AIR, SURFACE and SUBSURFACE.

G 4.3.1.1: Initial Category Assignment

The system shall assign a category to all new vehicular tracks on initial entry. The initial category assignment shall be based on category speed as follows:

- speed less than category speed: SURFACE;
- speed greater category speed: AIR.

The category speed shall be user modifiable, the preset value shall be 50 knots.

G 4.3.1.2: Category Change Processing

The user shall be provided with the capability of manually changing:

- the category of all firm vehicular tracks;
- category of a tentative track for New Track Establishment (Goal 4.2.1).

G 4.3.2: Track Identity Determination

The system shall perform a vehicular track identity determination. Track identities are UNKNOWN, FRIEND and HOSTILE.

G 4.3.2.1: Initial Vehicular Track Identity Assignment

The system shall assign a vehicular track the identity UNKNOWN upon new track establishment (Goal 4.2.1).

G 4.3.2.2: Vehicular Track Identity Change Processing

The user shall be capable of changing the identity of all vehicular tracks .

G 4.4: Search and Detection Function

The Search and Detection Function is the search for, detection and entry of vehicular tracks by ownship sensors. Ownship sensors for LCCDS are assumed to be search radars only. This goal represents the radar interface (TBD) and auto tracking capability of Increment four.

G 4.4.1.: Radar Interface

The system has to provide external interfaces to search/navigational radar systems. Since the characteristics of the radar are not yet known, a detailed refinement cannot be provided at this time and shall be part of further research.

G 4.4.2.: Interface Radar and OODBMS

The system has to provide an internal interface between the OODBMS and the radar. The information passed from radar to OODBMS shall result in the instantiation of objects of class TENTATIVE TRACK and initiate new track establishment as described in Goal 4.2.1 or shall be used to update the positions of existing objects as described in Goal 4.2.2.1.

G 4.5: Communication Function.

The system shall support the establishment and maintenance of digital communications.

G 4.5.1: Link 11

Implement an external communication function as described in Increment five based on a TBD classified Interface Design Specification.

G 4.6: Tableau System

The LCCDSWS shall provide detailed information on all aspects of the tactical situation and system control, operating parameters and status. This information shall be indexed for easy user access and presented in tableau form using Alphanumeric I/O windows.

These "tableaux" of information shall be directly accessible by the user, or be automatically displayed as a result of some program action. In any case, these tableaux should be context sensitive, providing a lower level of detail on demand or as a support function of some related process.

G 4.6.1: Basic Features

The purpose of the Tableau system is to provide the user with the detailed information he sometimes needs without cluttering or confusing the tactical picture presented on the TacPlot.

One important feature of the Tableau system is to provide a formatted, well organized summary of functions defined by the user via some complex, and possibly lengthy, series of menu choices and Alphanumeric inputs. A prime example of this situation is the Display Doctrine. It may be a complicated process to create Display Doctrine statements involving very rigidly structured sequences of menu options and dialog. The user must be able to review what he's built, and certainly refer to it later, in some direct and efficient manner.

G 4.6.2: Tableau Representation

Tableaux shall be represented using Alphanumeric I/O windows. These windows shall have the resize, reposition and scroll capability as outlined in the MMI conventions (Goal 1.1.1.2). However, these functions, as well as the capability for user input data directly into a tableau, should be considered on a case-by-case basis depending on the structure and content of each.

G 4.6.3: Tableau Types

There is a large body of tactical reference material which does not require continuous display, but should be quickly available to the user. Much of the information currently maintained on plexiglass "grease boards" fits into this category and should be maintained on the LCCDS in tableau form.

In addition to the default tableaus provided with the system, the user shall have the capability of defining new ones for specific purposes and adding them to the Index tableau. User defined tableaus shall be created using menu choices and fill-in-the-blank templates in much the same style as the Display Doctrine constructors (Goal 1.3).

The following default types of information tableaus are provided to further describe the functionality and concept of use. They shall be considered a minimum set.

G 4.6.3.1: Index

The Index tableau shall provide a list of all default and user-defined tableaux. This index shall be represented in a standard support window format (Goal 1.1.1.2.1.2.4) using one of the scroll features to scan the list using the Trackball pointer. The user should be able to move to the desired area of the list using the scroll bar and then highlight (by pointing with the trackball) to the tableau to be displayed. A conceptual example of this window is provided in Figure 27.

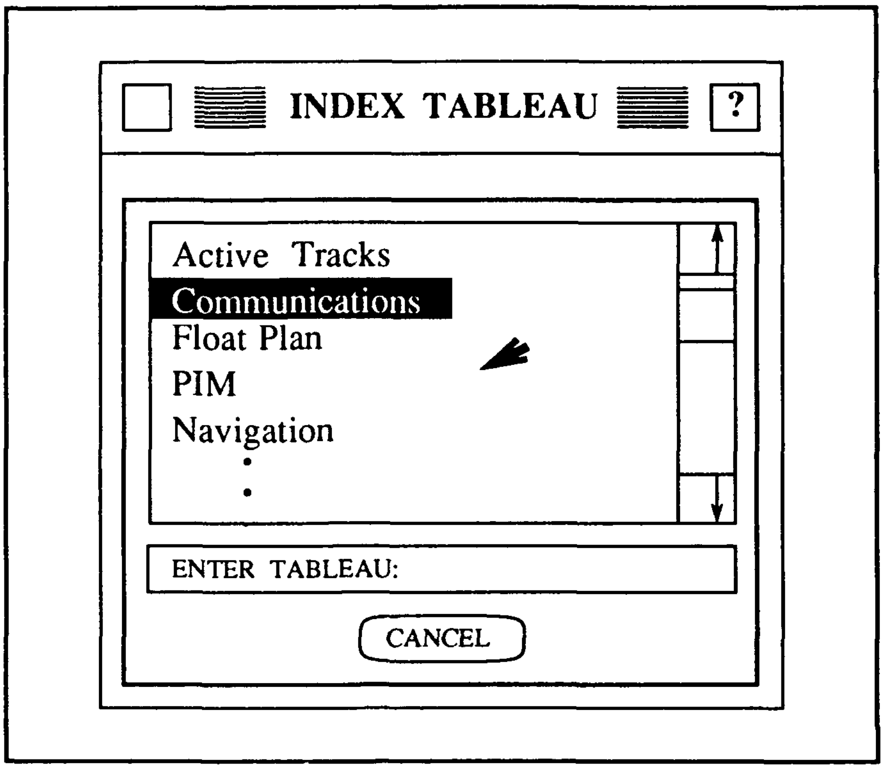


Figure 27: Index Tableau Support Window

G 4.6.3.2: Navigation

The Navigation tableau shall provide detailed information on ownship navigation including, but not limited to, the following:

- time (standard GMT and local format - hh:mm:ss)
- lat/long position (dd:mm:ss)
- course (relative to true and magnetic north - 000.0 T/M)
- speed (000.0 knots)
- magnetic variation (degrees East/West - 00.0 E/W)
- set (degrees true - 000.0 T)
- drift (speed in knots - 000.0)
- wind direction (degrees true - 000.0 T)
- wind speed (speed in knots - 000.0)

- distance to next waypoint (NM great circle or rhumb (direct) distance)
- ETA next waypoint (DTG format - 212030Z JUN 90)
- ETE next waypoint (hh:mm:ss)
- Navigation source (inertial, SATNAV, Omega, DR, etc.)

G 4.6.3.3: PIM

The PIM tableau shall maintain continuous status of Ownship progress along a selected steaming route defined in the Float Plan tableau. This tableau shall provide the following information:

- PIM course and speed
- current PIM position
- Ownship distance ahead/behind PIM
- ETE to PIM (based on current or user entered trial course/speed)

G 4.6.3.4: Float Plan

This tableau provides the user a tool for defining up to six steaming routes of up to 50 waypoints each. Each waypoint is defined by a latitude (dd:mm:ss N/S) and longitude (dd:mm:ss E/W). A steaming route shall consist of a numbered list (in order of entry) of waypoints. The user shall, at any time, be allowed to edit this tableau by selecting the desired steaming route and modifying, adding or deleting one or more of its associated waypoints.

It is highly recommended that the content and format of the Float Plan steaming routes follow the structure of the PIM portion of OPGEN ALPHA. This form is supported by, and described in detail in, the NCCS-A MMI specification [Ref. 29:pp. 75-79].

G 4.6.3.5: Communications

The Communications tableau, or set of related tableaux, shall provide all essential information regarding radio and link circuits, frequencies, crypto change-over times, daily call-signs for desired friendly units. Additionally, it should provide a menu of selectable message templates for various reports.

G 4.6.3.6: Active Tracks

The Active Tracks tableau shall provide a list of all tracks currently held in the Tactical Database along with time of last update. This list shall be represented in a standard support window format (Goal 1.1.1.2.1.2.4) using one of the scroll features to scan the list using the Trackball pointer. The user should be able to move to the desired area of the list using the scroll bar and then highlight (by pointing with the trackball) any track. Clicking on the highlighted track will display all the detailed information (from the Tactical Database) pertaining to it. This information shall be presented in the Selected Track tableau (Goal 4.6.2).

G 4.6.3.7: Selected Track

This tableau provides all the information held in the Tactical Database pertaining to the selected track (chosen from the Active Tracks tableau).

G 4.6.3.8: Display Doctrine

The Display Doctrine tableau shall enable the user to view all existing Display Doctrine statements. This tableau should be an Info object, allowing the user to view the data, but not change it. Changes should be made only through the Display Doctrine menu dialog sequence described in Goal 1.3.

G 4.6.3.9: Data Link Control

This tableau shall display Link 11 control parameters and amplifying information. The Link 11 control parameters shall include ownship track number (TN), DLRP lat/long, and TN pool. Amplifying information on other Link 11 participating units (PU's) including TN and platform type, shall also be provided.

G 4.6.3.10: System Status

The System Status tableau maintains the current operational status of all LCCDS software, hardware and external interfaces. This tableau shall be updated automatically based on program and user action regarding system initialization parameters, selected interfaces, LCCDS hardware configuration and availability and current state of the operational software.

An important feature of this tableau shall be the capability for the user to manually inhibit the function of any hardware, interface or software component based on mission requirements, operational status, etc. For instance, in an emission control environment (EMCON) the user shall have a method for disabling the Radar interface to preclude unauthorized or accidental radiation.

G 4.6.3.11: User Defined Tableaux

The system shall provide the user with the capability to define both static and dynamic tableaus. The specific content and usage shall be determined locally by the user (or user's command) and constructed in much the same manner as the Display Doctrine using "smart" templates and forms with appropriate ranges of available choices.

The differentiation between static and dynamic tableaux is the degree of complexity involved. Static tableaux contain information provided by the user and is not maintained, changed or updated by program action. Procedures, checklists, etc. are examples of the type of information available through static tableaux.

Dynamic tableaux are more complicated to define because it is intended that the program will somehow act upon or effect their contents. The purpose and functionality of dynamic tableaus shall be very carefully restricted, providing clear guidelines for the available range of user defined operation. The main source of on-line information to be made available for use in these tableaux shall be the Tactical Database.

G 5: Implementation in Steps

The system is supposed to be modularized to allow an implementation in incremental steps.

G 6: LCCDSWS is highly Concurrent

The system is supposed to be highly concurrent and prepared for future extensions. Possible extensions are described in Chapter V as evolution of the system.

G 6.1: Provide Real-Time Scheduling and Allocation

The system shall provide internal scheduling and allocation of the processor and resources to real time requirements.

G 6.2: Process Load Orders in Real-Time

The system shall process program load orders in real time.

III. LCCDS ESSENTIAL MODEL

This chapter provides a high level structural analysis of the LCCDS using the Essential (Yourdon) model [Ref. 18]. Yourdon's Essential model attempts to define *what* the system must do in order to satisfy the user's requirements while avoiding any implementation level details. It is a logical, conceptual model of the system.

The Essential model is comprised of two basic components; the Environmental model and the Behavioral model. Yourdon's Environmental model has some overlap with the goal tree component of Berzins' model (used for the informal requirements analysis provided in Chapter Two). Both attempt to define the boundary between system and external world. The major distinction between the two is that Yourdon's Environmental model includes more detail on external communications requirements in the form of an Event List.

The Behavioral model tries to define the internal processes and data objects necessary for supporting the "events" interacting with the external world. Events are defined as external stimuli to which the system must somehow respond [Ref. 18:p. 340]. The Behavioral model is a lower-level definition of the system, utilizing data flow diagrams, entity-relationship diagrams, state-transition diagrams, data definition (dictionaries) and process specifications.

The Essential model therefore provides an important link between informally stated user desires and eventual functional capabilities of the system. It provides a tool for the system analyst to model the desired system at a level high enough to be understandable to the user and still provide the necessary guidance for implementation.

A. The Environmental Model

1. The Statement of Purpose

The purpose of the project reported here is to develop the prototype of a software system for a Low Cost Combat Direction System (LCCDSWS) that implements the basic features of a Combat Direction System on a commercially available microprocessor based workstation.

The purpose of the LCCDS software system is to maintain and display a real-time tactical picture of a naval vessel's external environment. This includes generating, tracking and managing all air, surface and subsurface tracks for a given geographic area, Special Points and shoreline maps.

2. The Context Diagram

The context diagram pictured below in Figure 28 is identical to the initial

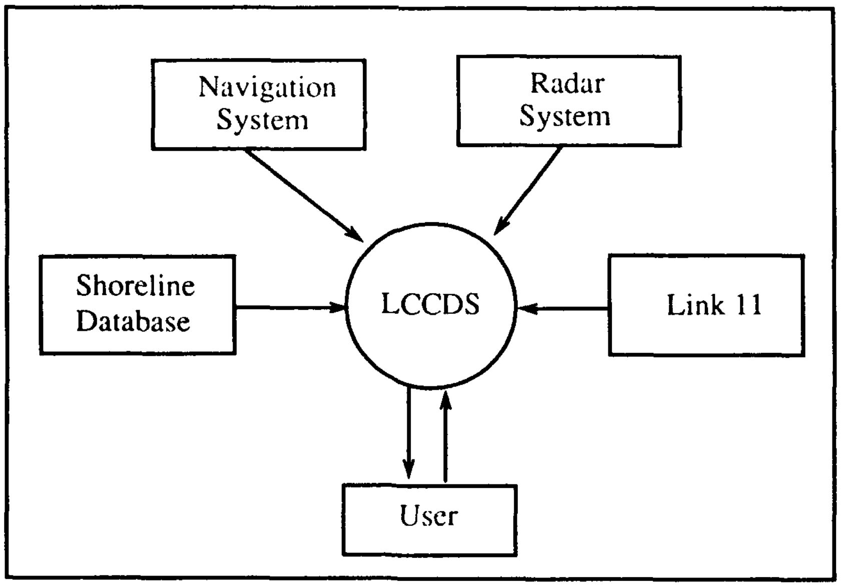


Figure 28: The Context Diagram (from Figure 10, page 21)

context diagram in Figure 10 and is provided here for convenience. The meaning and use of the context diagram is the same for both the Berzins and Yourdon models.

3. The Event List

The event list is an informal description of the events which take place in the external environment to which the LCCDS must respond. Events are defined from the perspective of the external world, i.e., from the outside looking in [Ref. 18:p. 351] .

1. Operator receives request for system initialization.
2. Operator sends system initialization parameters (Goal 2.1).
3. System displays TacPlot (Goal 1.2.1).
4. TacPlot displays ownship symbol(Goal 1.2.1.1 (7)).
5. TacPlot is refreshed corresponding to ownship position (Goal 4.1.1).
6. Operator enables interface to navigation system (Goal 2.2.1).
7. Navigation system sends ownship messages every 3 minutes.
8. Operator disables interface to navigation system (Goal 2.2.1).
9. Operator enables interface to radar system (Goal 2.2.1).
10. Radar system sends radar messages.
11. TacPlot displays radar track symbols.
12. Radar track symbols are repositioned or updated on Tacplot (Goal 4.2.4).
13. Operator disables interface to radar system (Goal 2.2.1).
14. Operator enables interface to Link 11 system (Goal 2.2.1).
15. Link 11 system sends Link messages.
16. Link 11 track symbols are displayed on TacPlot.
17. Link 11 tracks symbols are repositioned on TacPlot.
18. Operator disables interface to Link 11 system (Goal 2.2.1).

19. Operator enables interface to shoreline database.

20. Operator sends map request.

21. Shoreline database accepts map request.

22. Shoreline database sends map message.

23. TacPlot displays map.

24. Operator generates new manual track (Goal 4.2.1).

26. TacPlot displays manual track symbol.

27. Manual tracks are repositioned or updated on TacPlot.

25. Operator updates position of a track (Goal 4.2.2.1).

26. Tacplot displays updated track at new position with new course and speed.

26. Operator repositions a track (Goal 4.2.2.2).

27. TacPlot displays repositioned track at new position with old course and speed.

28. Operator enters new course and speed for a track. (Goal 4.2.3.2).

29. TacPlot displays track at old position with new course and speed.

30. Operator terminates a track (Goal 4.2.7.2).

31. The selected track is not anymore displayed on the TacPlot.

32. Operator changes category of a track (Goal 4.3.1.2).

33. TacPlot displays track symbol with changed category.

34. Operator changes track identity (Goal 4.3.2.2).

35. TacPlot displays track symbol with changed identity.

36. Operator changes range of TacPlot.

37. The picture of the current situation is updated according to the range selected.

38. The operator hooks a position on the Tacplot.

39. The operator is requested to generate a new track or special point when neither of the current tracks is at the hooked position.

40. Additional information is shown/further action requested when an already existing track is at the hooked position.

B. The Behavioral Model

The Behavioral model describes the internal behavior required of the system to satisfactorily interact with the external world. For the purposes of this research, only the data flows, data structures and data dictionary for the LCCDS are defined. We used a CASE tool, *Software Through Pictures*, which supports the Yourdon model and associated symbology to create the above mentioned components of the Behavioral model.

1. Data Flow Diagrams

Data flow diagrams are commonly used systems modeling tools whose purpose is to graphically describe a system as a group of functions, processes or objects in a network configuration related by "data flows" connecting the various processes to each other [illegible] the external world. The data flow diagram, however, presents only one perspective of the system, its functions and their associated data objects. Data flow diagrams, or DFD's, cover a large range of system detail, from the high level concepts, to low level processes and system internal architecture. We provide the first two levels, Level Top and Level Zero.

Level Top is more detailed version of the Context diagram (see Figure 29). It includes a description of the data objects which must pass between LCCDS and the external environment. Level Zero (Figure 30) is one level of detail lower, explaining the major system processes and data objects necessary for supporting LCCDSWS functionality.

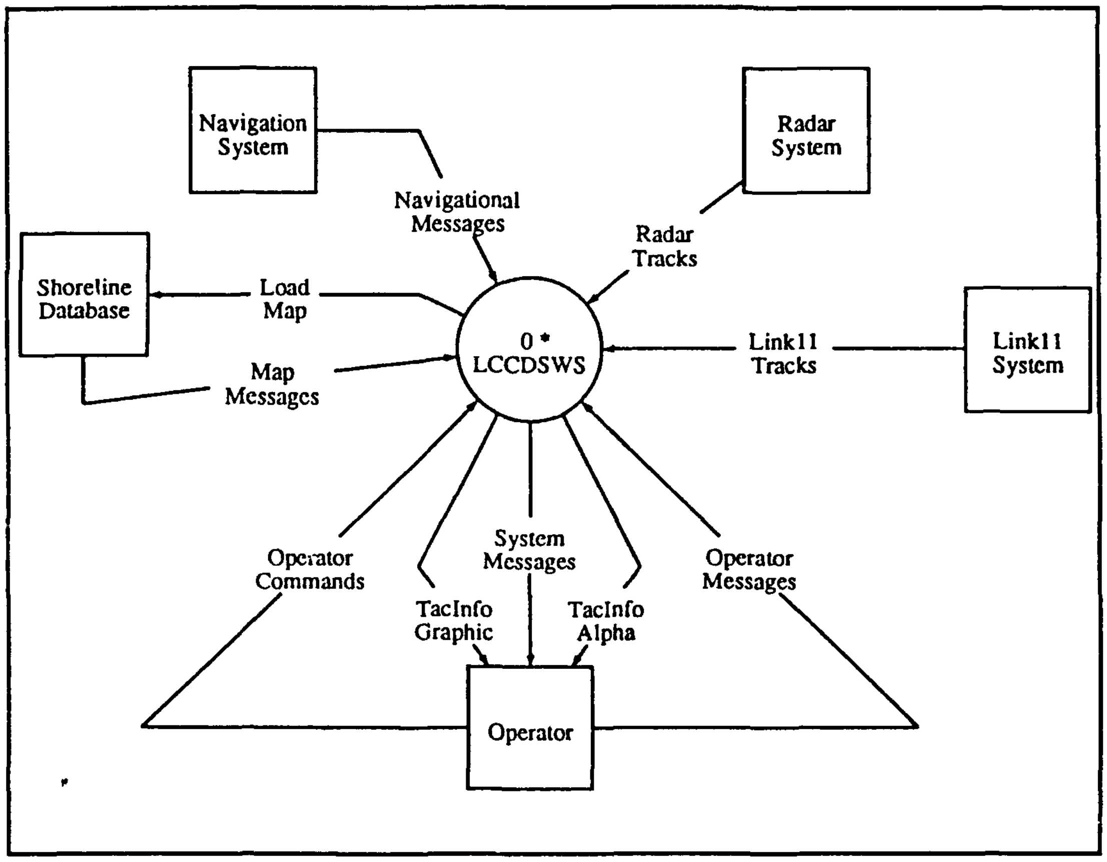


Figure 29: The Top Level Diagram

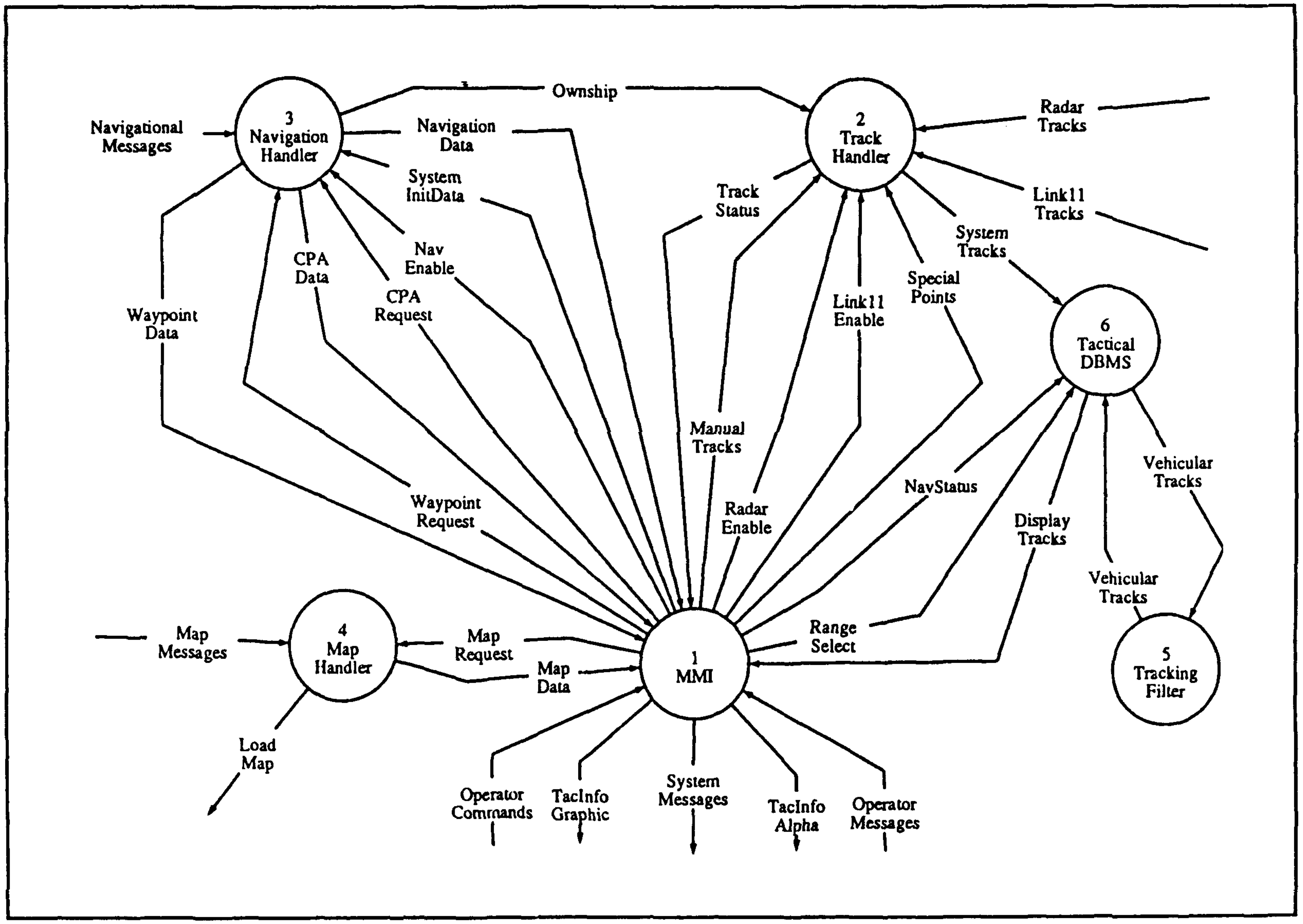


Figure 30: Level Zero Data Flow Diagram

2. Data Structure Diagrams

Hierarchical data structure diagrams are used to decompose data flows and data stores defined in the data flow diagram. Data structure diagrams allow users to view the system from a "data" perspective. For this reason we provide in the following the data structures for the level 0 data flow diagram as generated with the Data Structure Editor (DSE) of Software through Pictures (StP). DSE is an interactive graphical tool for drawing data structures. It can generate declarations for C, Ada, and Pascal, as well as input to the StP Data Dictionary system (provided by the next section). StP generated all the data structure diagrams provided in the following series of figures (Figures 31 - 69).

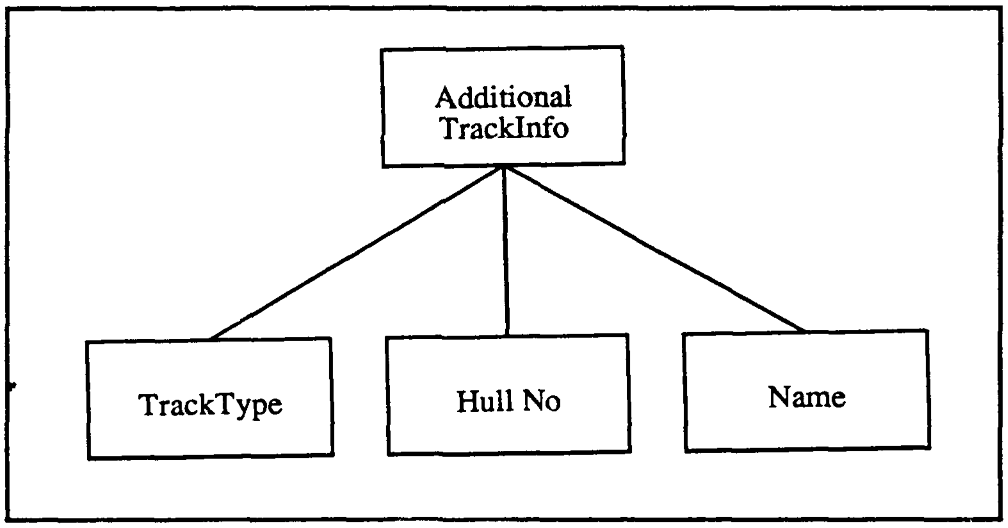


Figure 31: Data Structure Diagram - Additional TrackInfo

These diagrams are crucial for clear definition of the data flows depicted in Figures 29 and 30. They represent high level objects, and structures of objects. Each data structure provided in each figure directly corresponds to a data flow or a component of a data flow of the Top or Zero level data flow diagrams shown in Figures 29 and 30. For ease of reference the diagrams are given in alphabetical order.

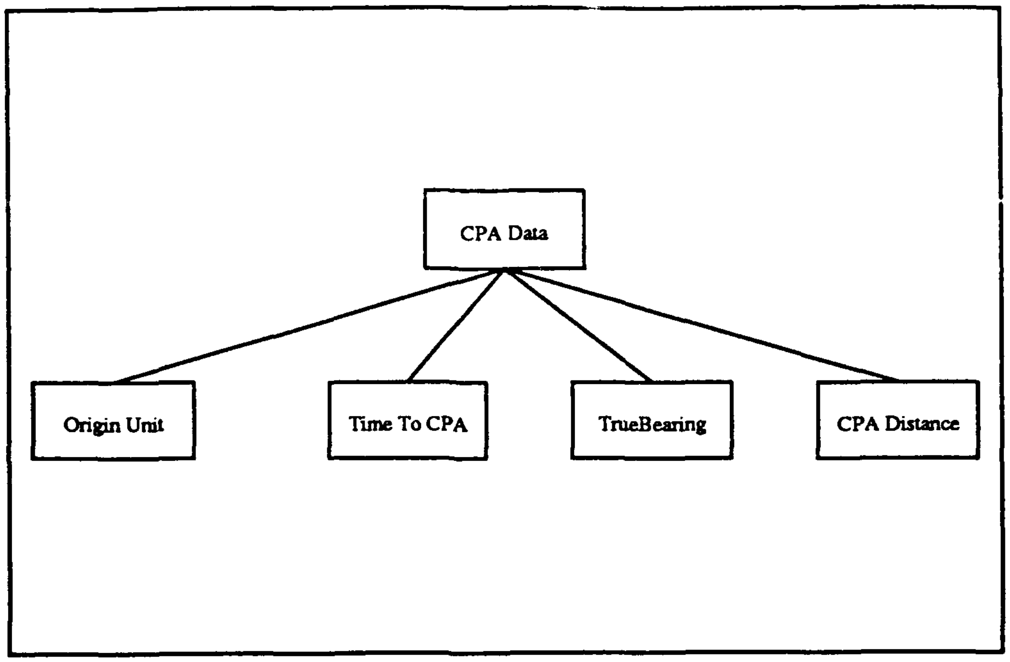


Figure 32: Data Structure Diagram - CPA_Data

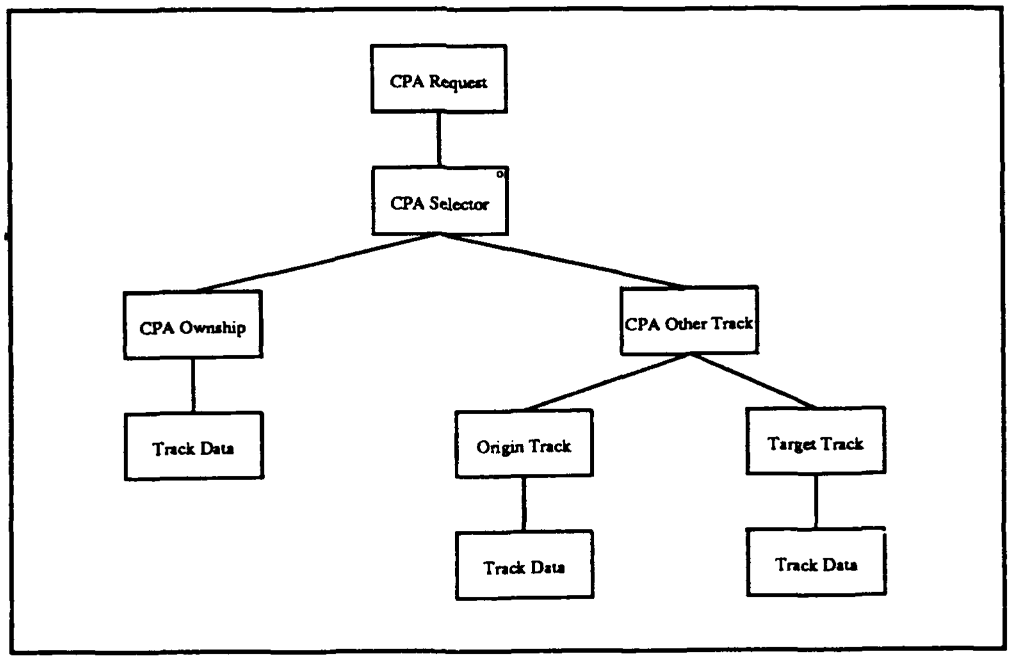


Figure 33: Data Structure Diagram - CPA_Request

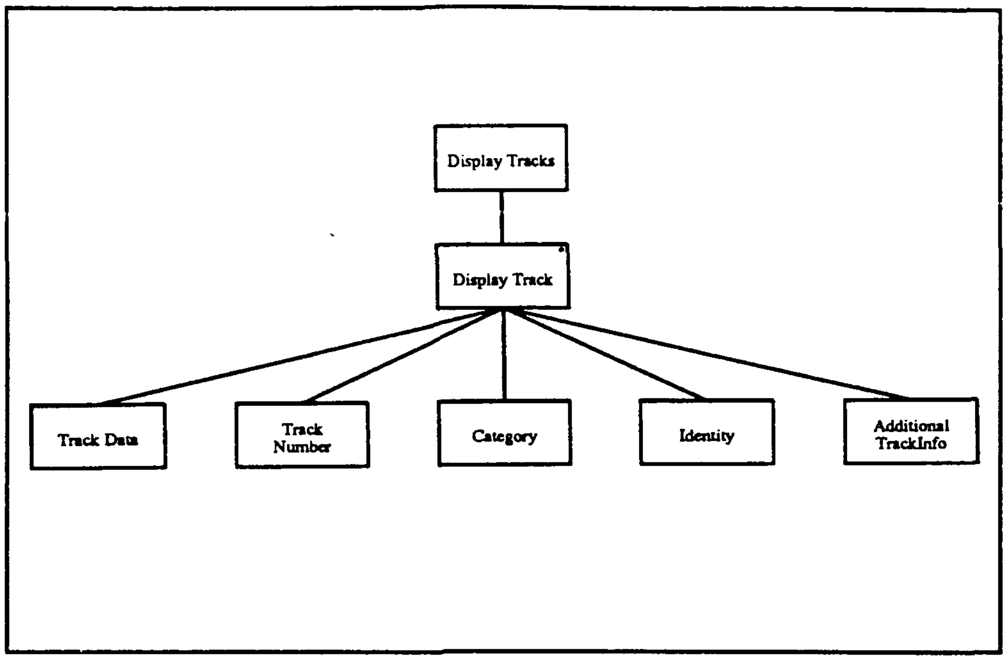


Figure 34: Data Structure Diagram - Display_Tracks

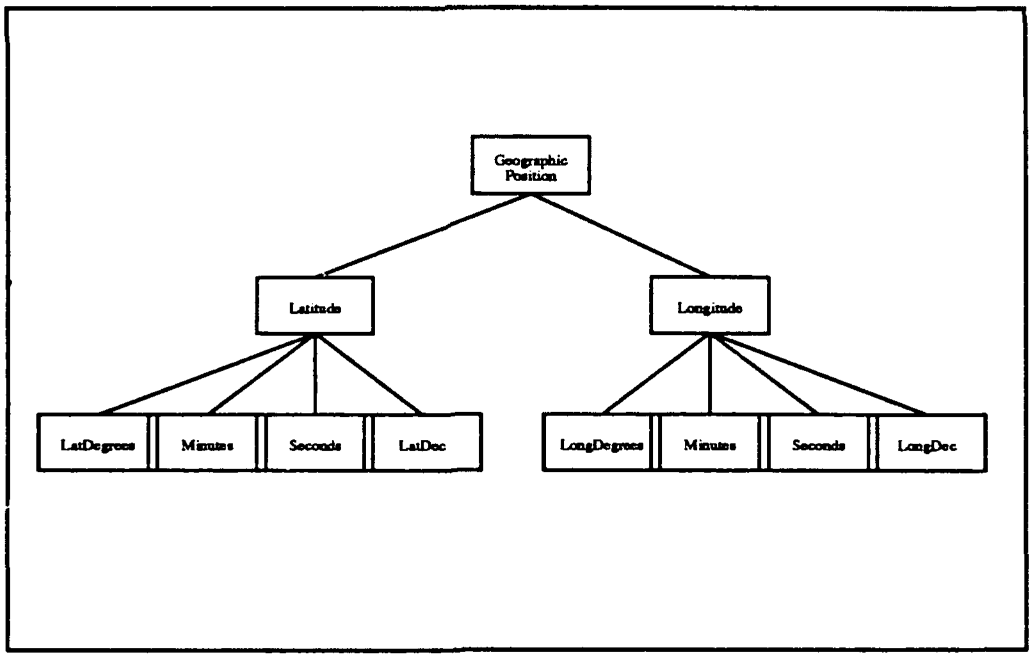


Figure 35: Data Structure Diagram - Geographic_Position

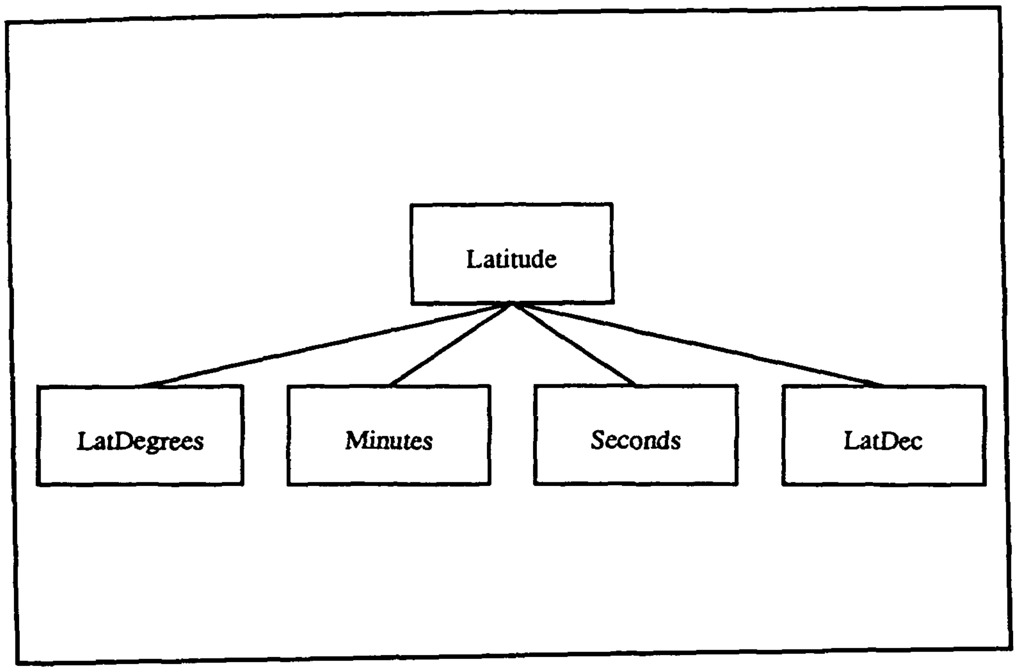


Figure 36: Data Structure Diagram - Latitude

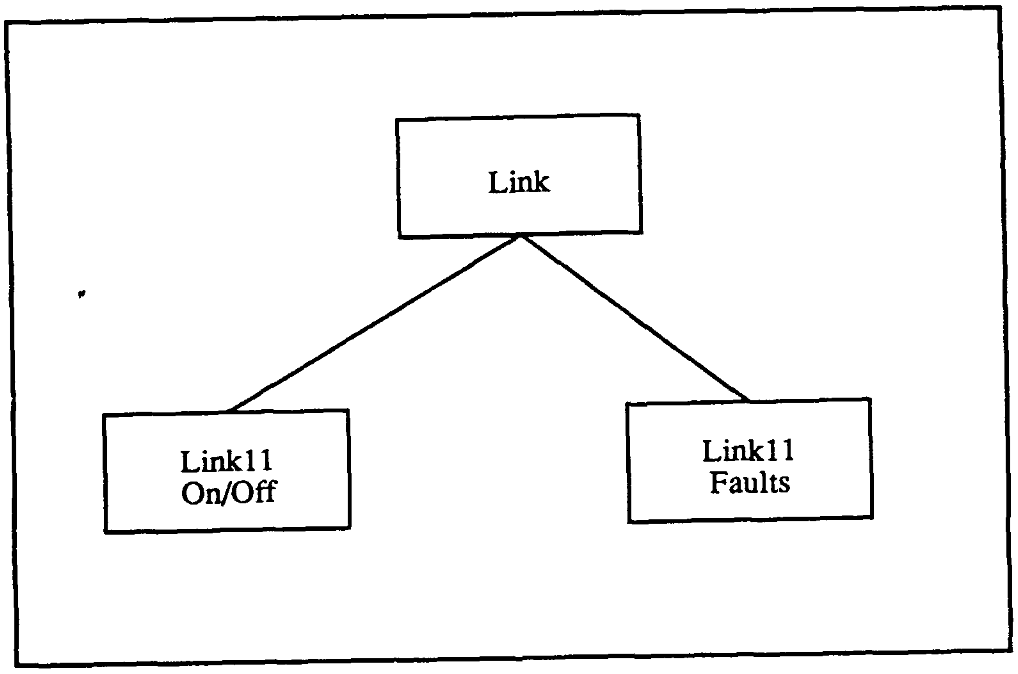


Figure 37: Data Structure Diagram - Link

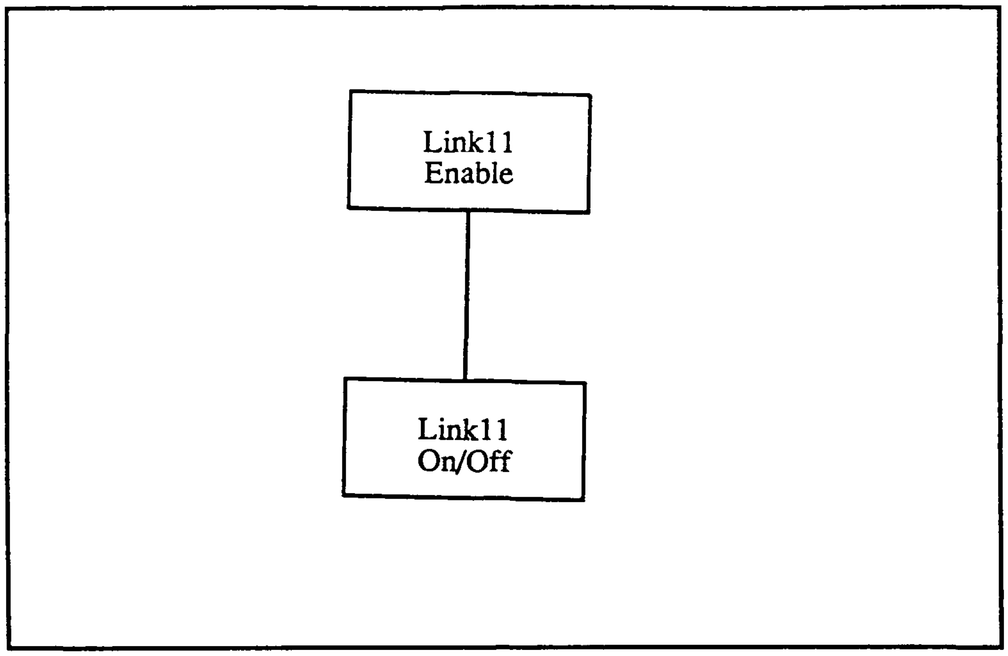


Figure 38: Data Structure Diagram - Link11_Enable

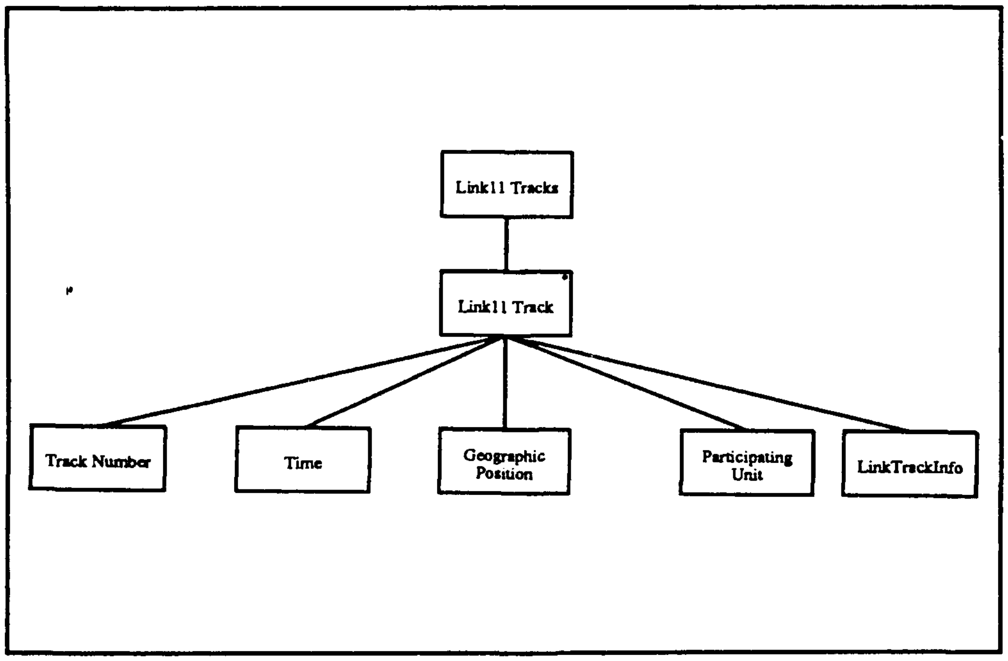


Figure 39: Data Structure Diagram - Link11_Tracks

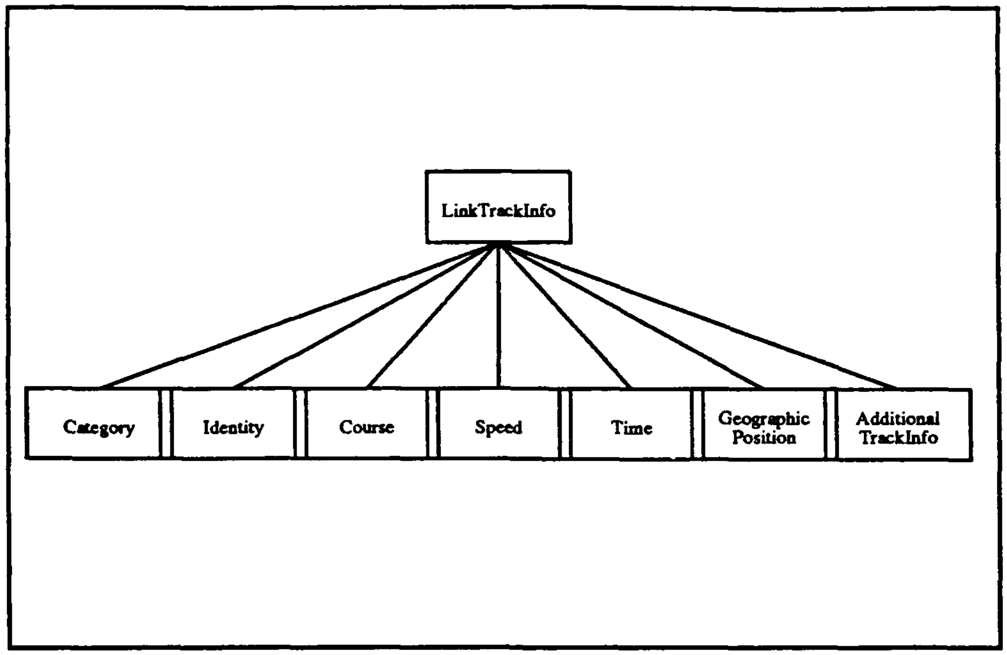


Figure 40: Data Structure Diagram - LinkTrackInfo

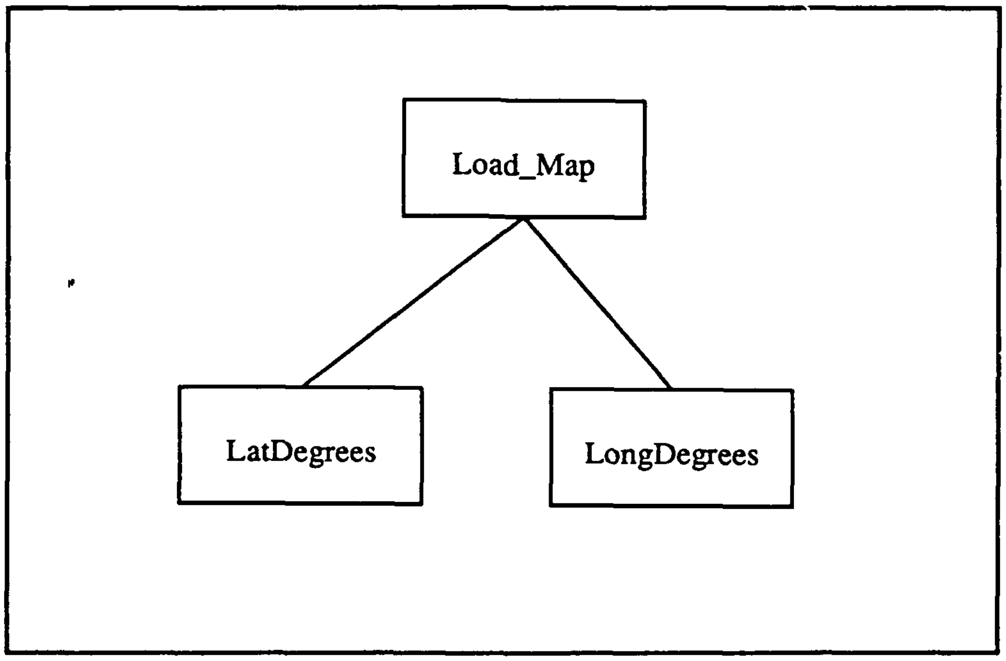


Figure 41: Data Structure Diagram - Load_Map

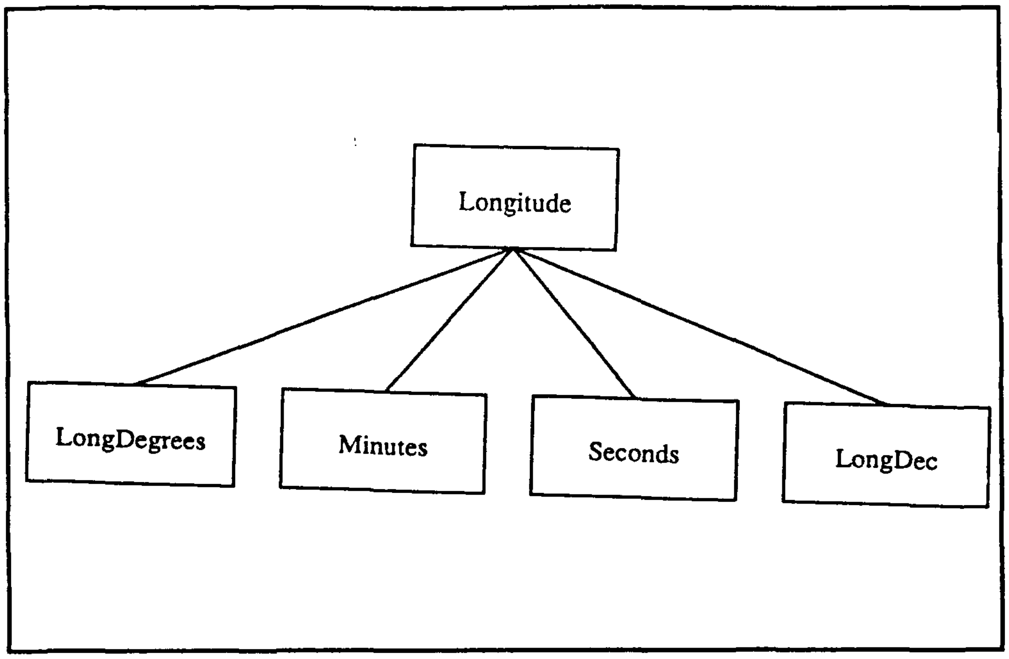


Figure 42: Data Structure Diagram - Longitude

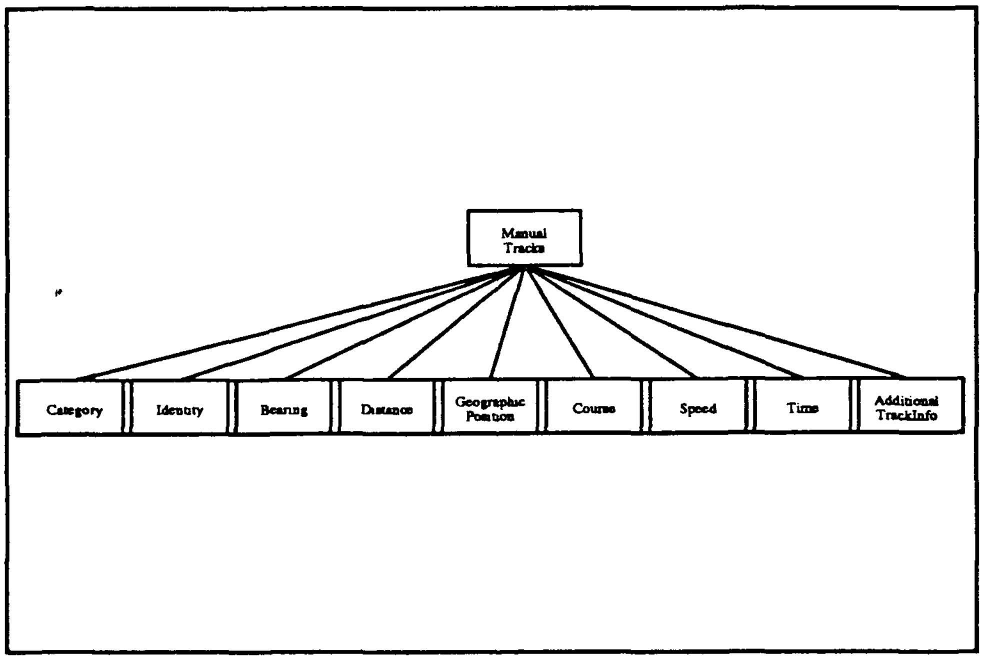


Figure 43: Data Structure Diagram - Manual_Tracks

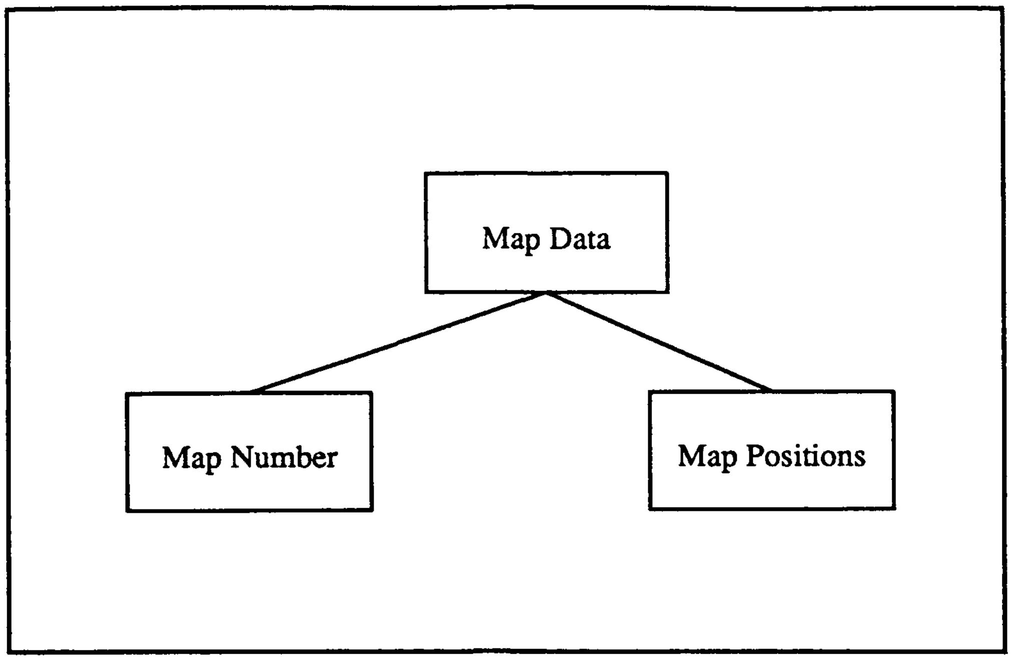


Figure 44: Data Structure Diagram - Map_Data

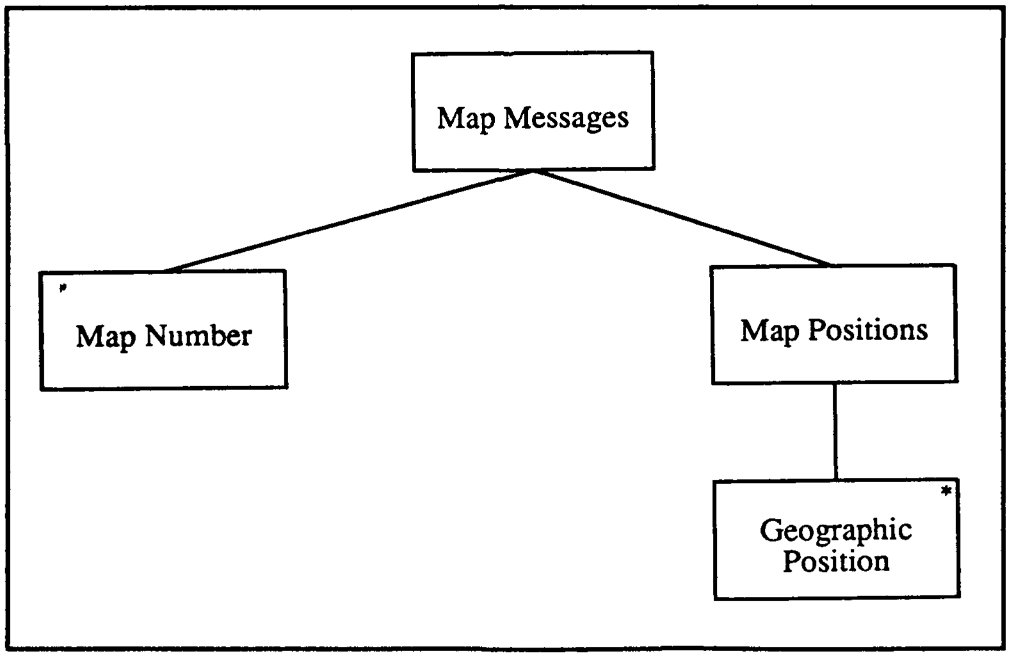


Figure 45: Data Structure Diagram - Map_Messages

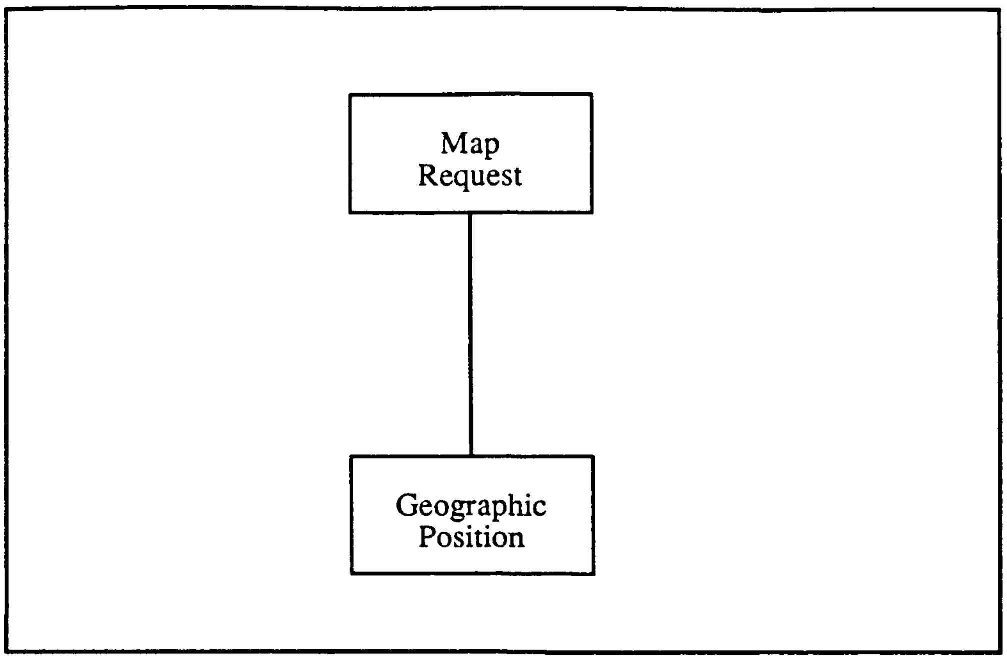


Figure 46: Data Structure Diagram - Map_Request

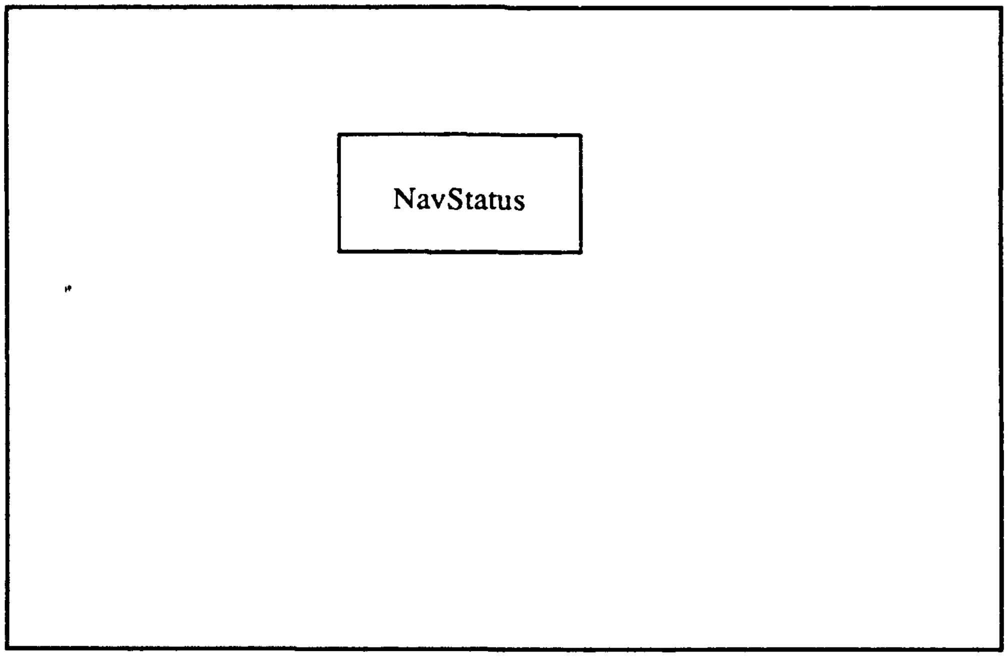


Figure 47: Data Structure Diagram - NavStatus

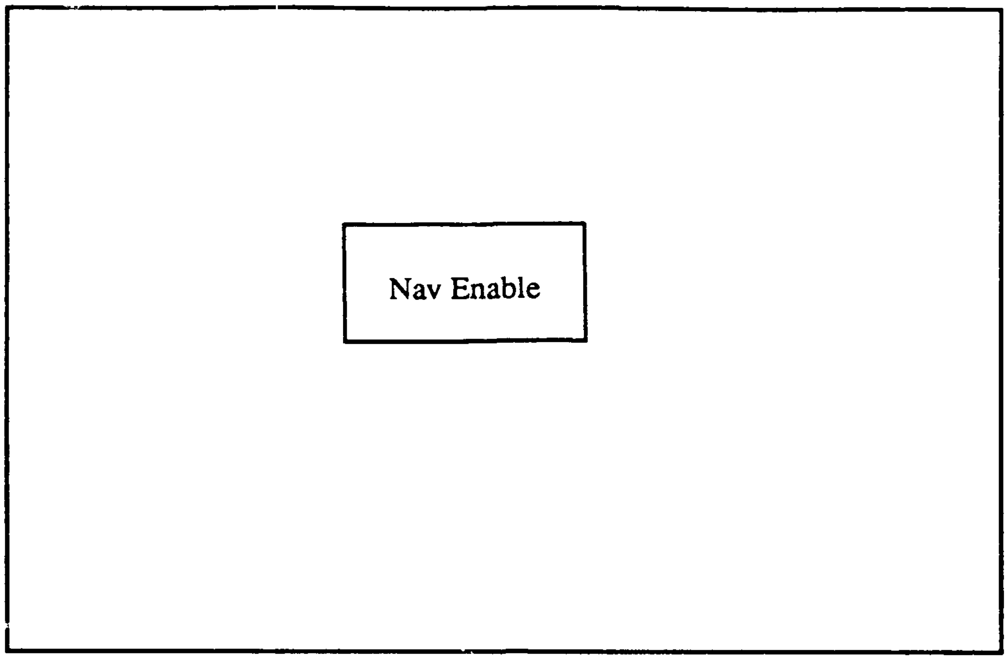


Figure 48: Data Structure Diagram - Nav_Enable

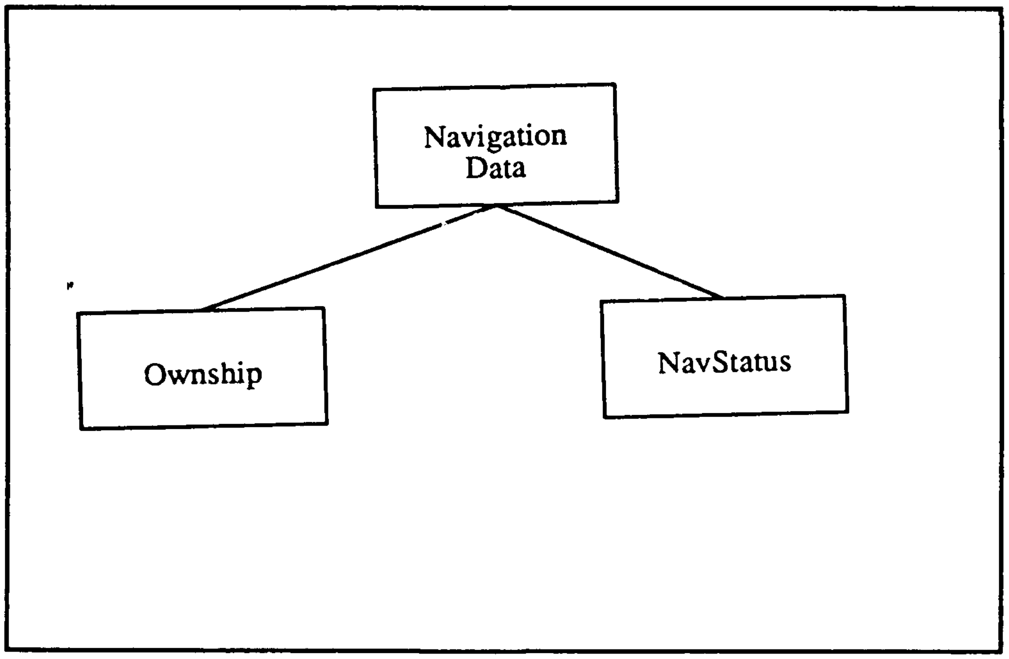


Figure 49: Data Structure Diagram - Navigation_Data

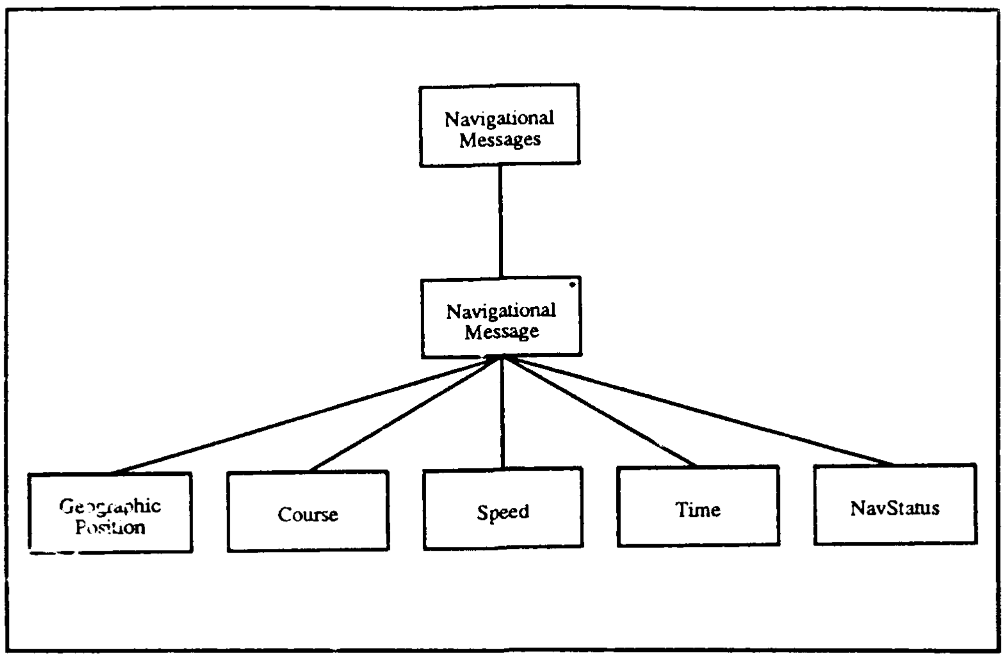


Figure 50: Data Structure Diagram - Navigational_Messages

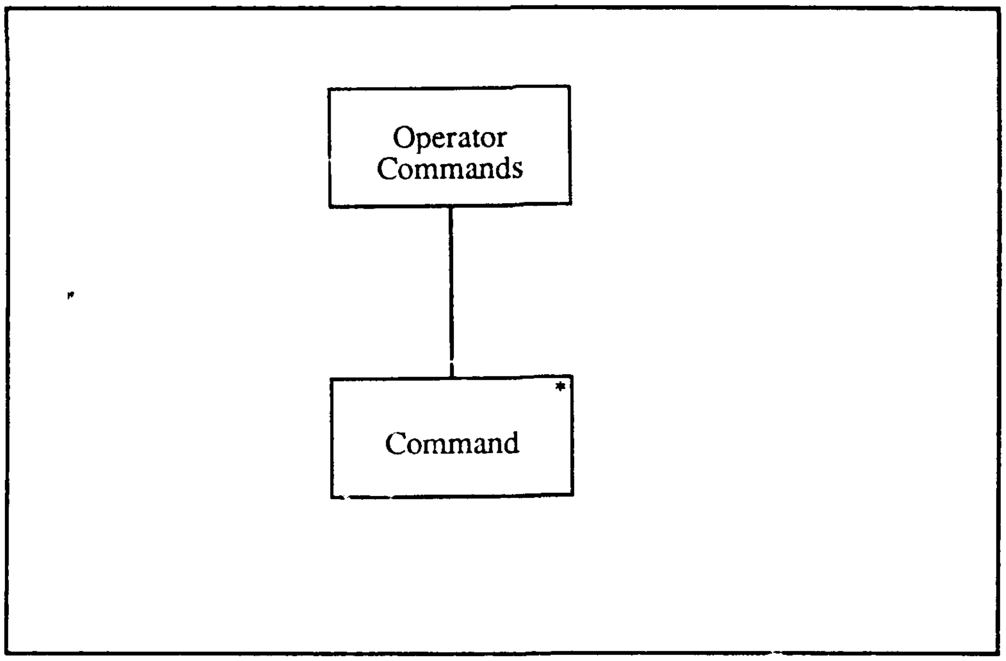


Figure 51: Data Structure Diagram - Operator_Commands

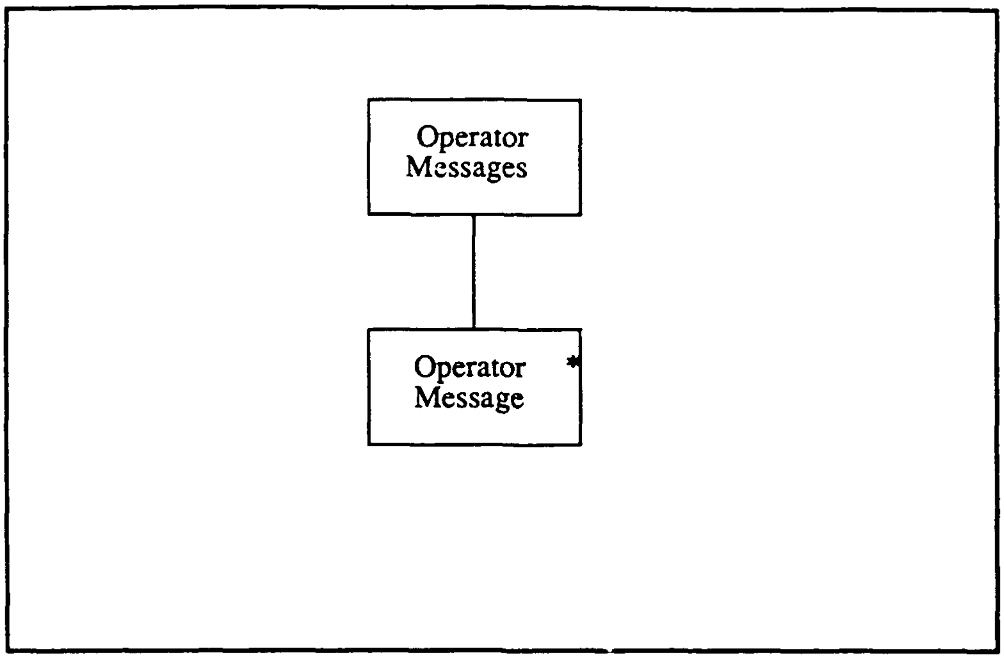


Figure 52: Data Structure Diagram - Operator_Messages

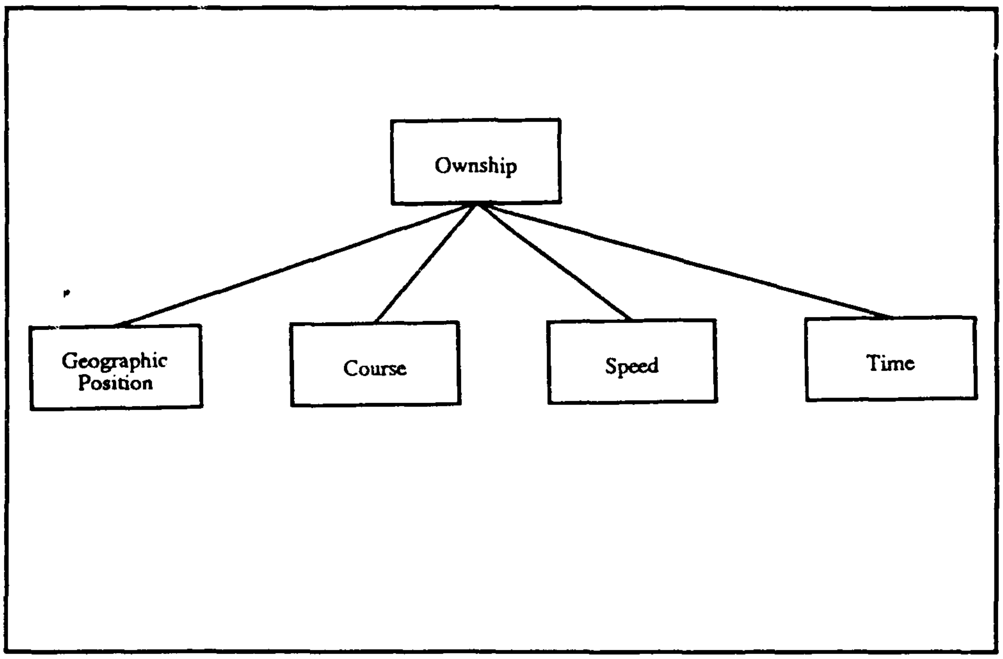


Figure 53: Data Structure Diagram - Ownship

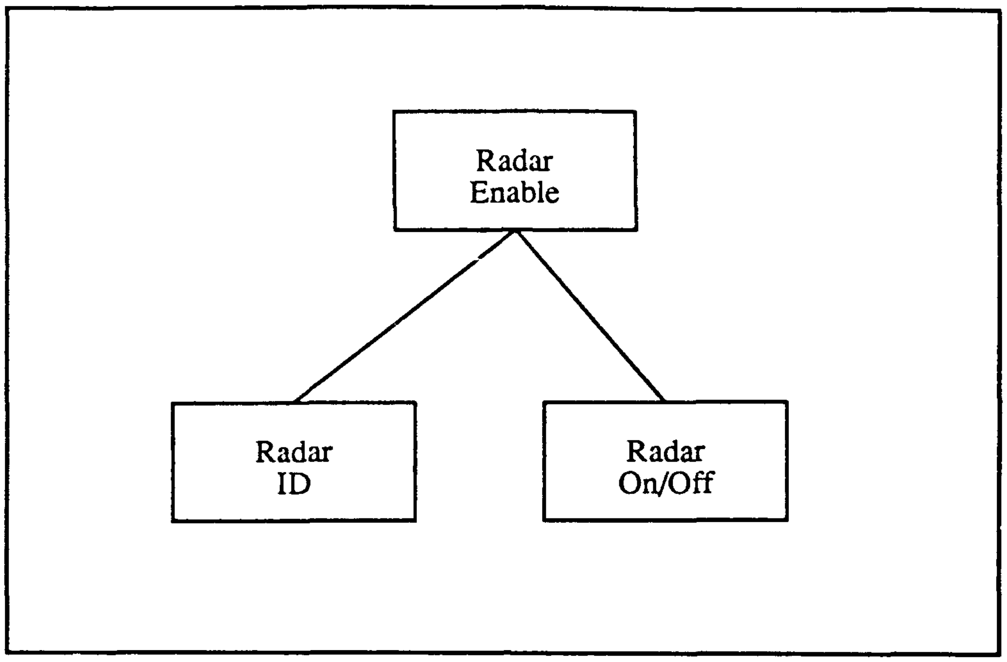


Figure 54: Data Structure Diagram - Radar_Enable

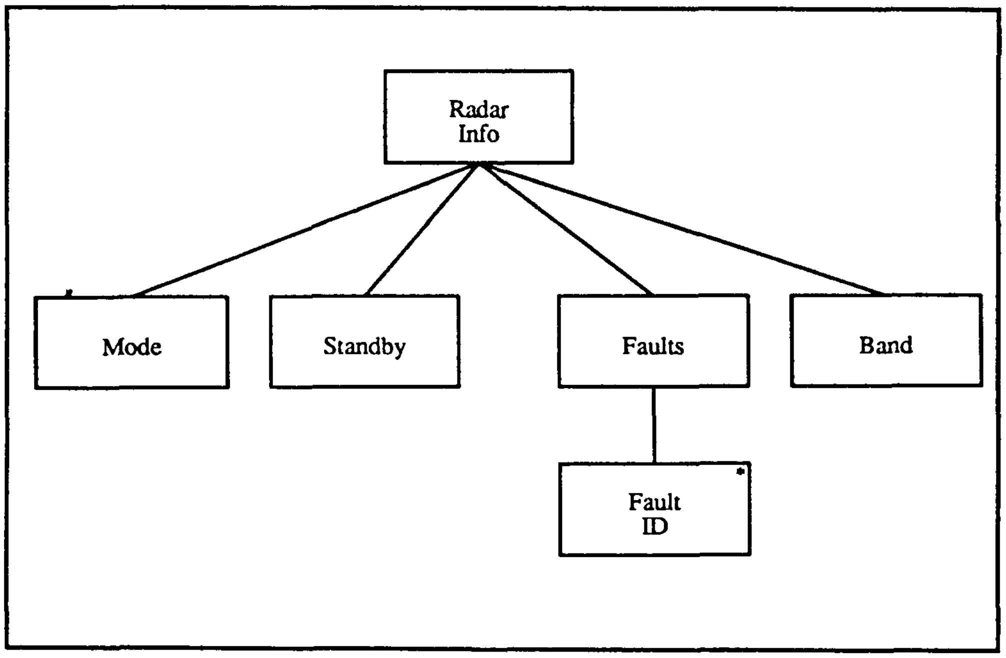


Figure 55: Data Structure Diagram - Radar_Info

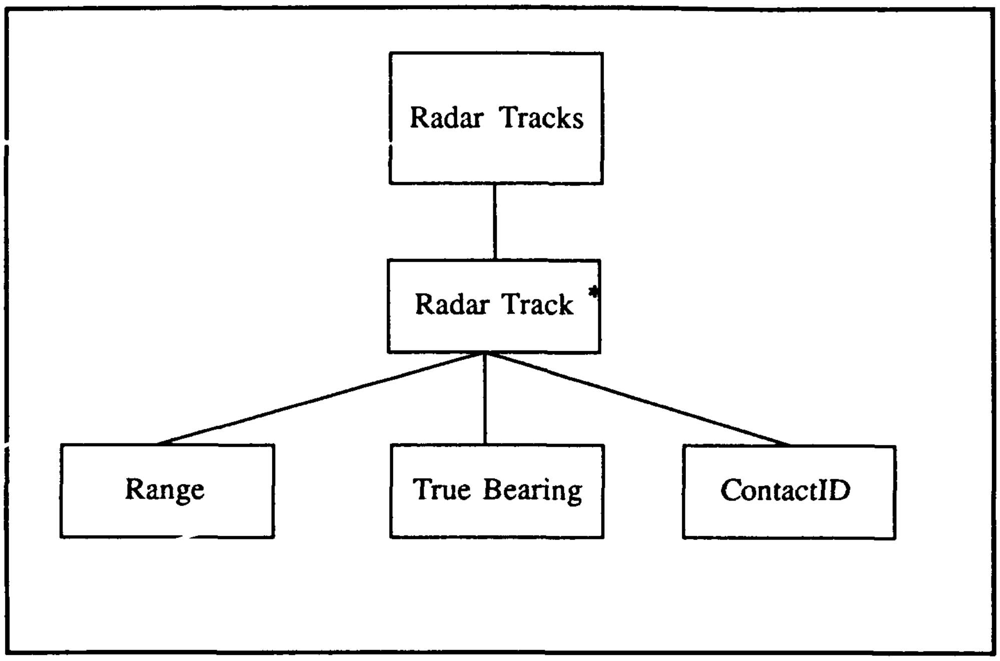


Figure 56: Data Structure Diagram - Radar_Tracks

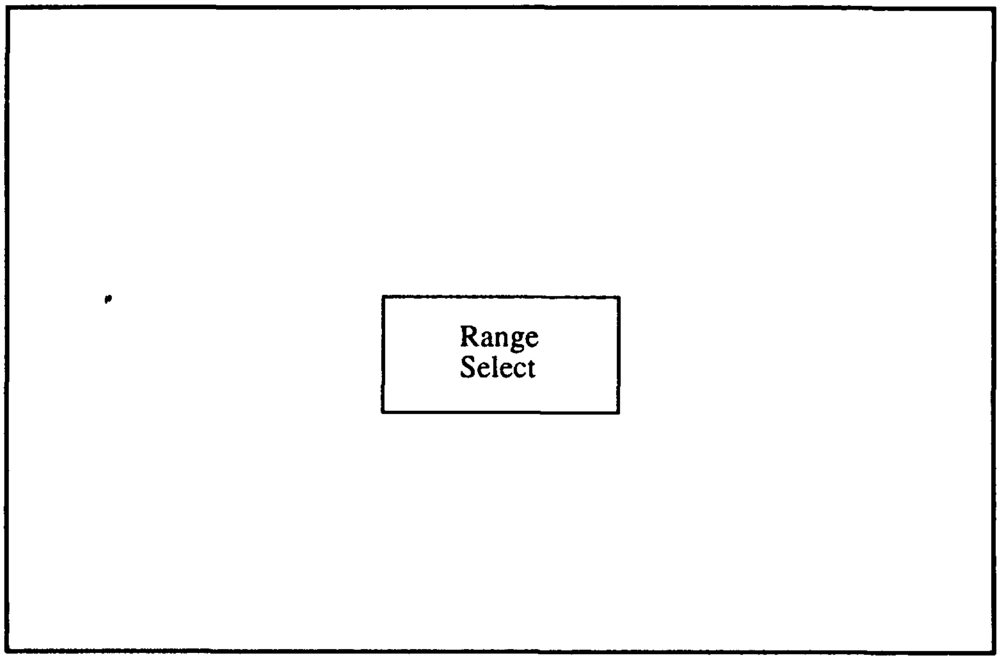


Figure 57: Data Structure Diagram - Range_Select

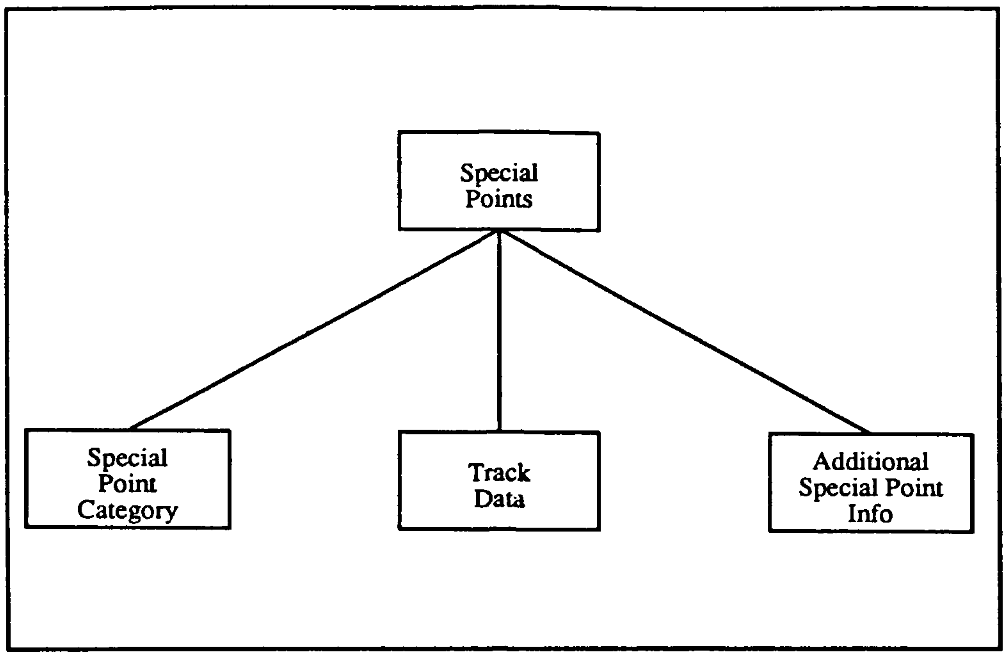


Figure 58: Data Structure Diagram - Special_Points

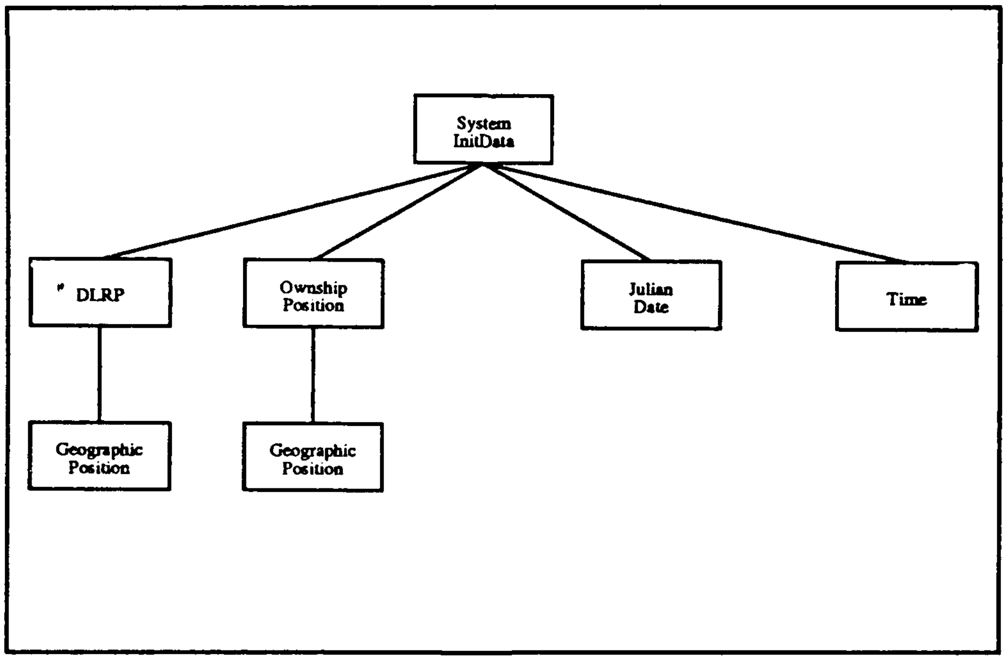


Figure 59: Data Structure Diagram - System_InitData

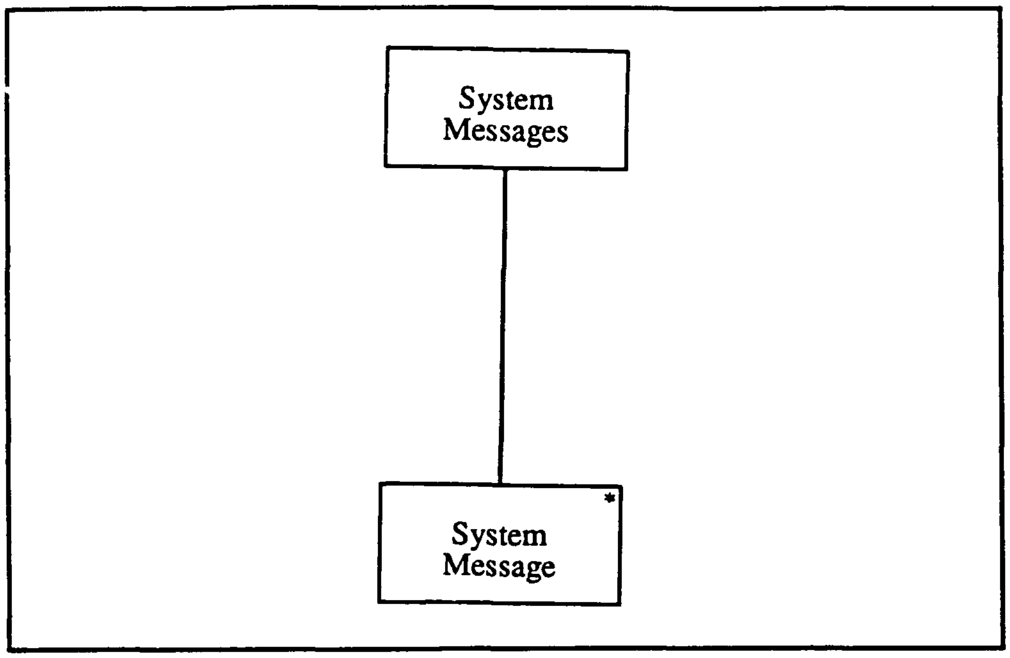


Figure 60: Data Structure Diagram - System_Messages

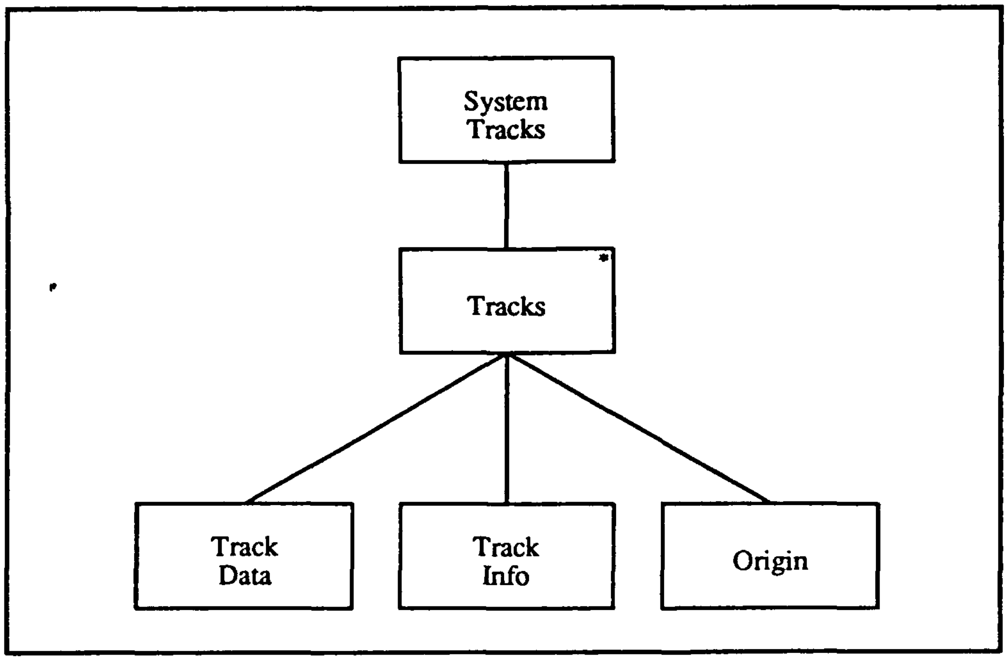


Figure 61: Data Structure Diagram - System_Tracks

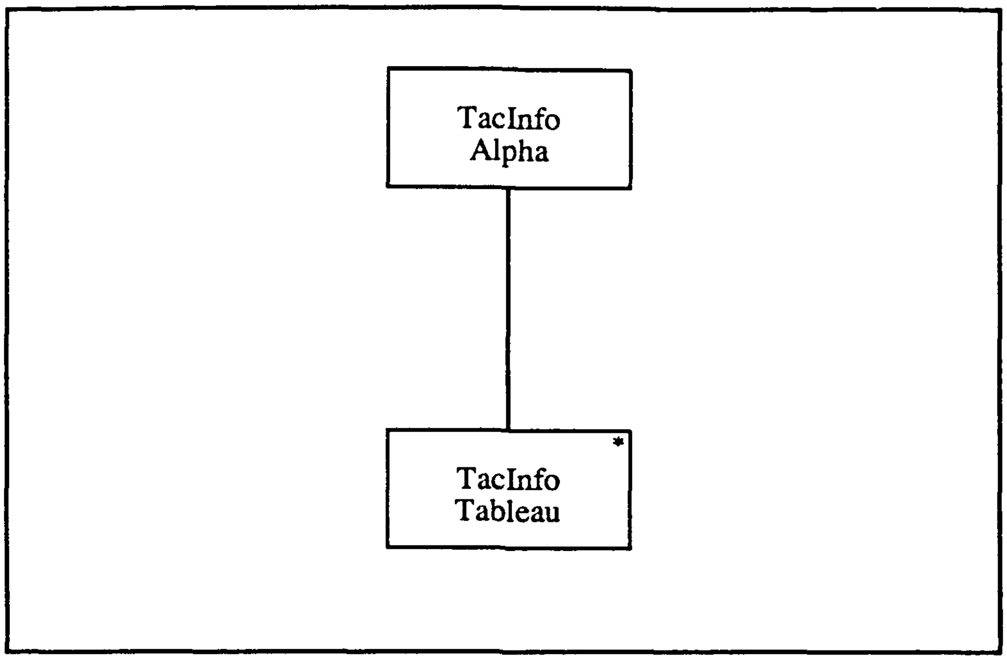


Figure 62: Data Structure Diagram - TacInfo_Alpha

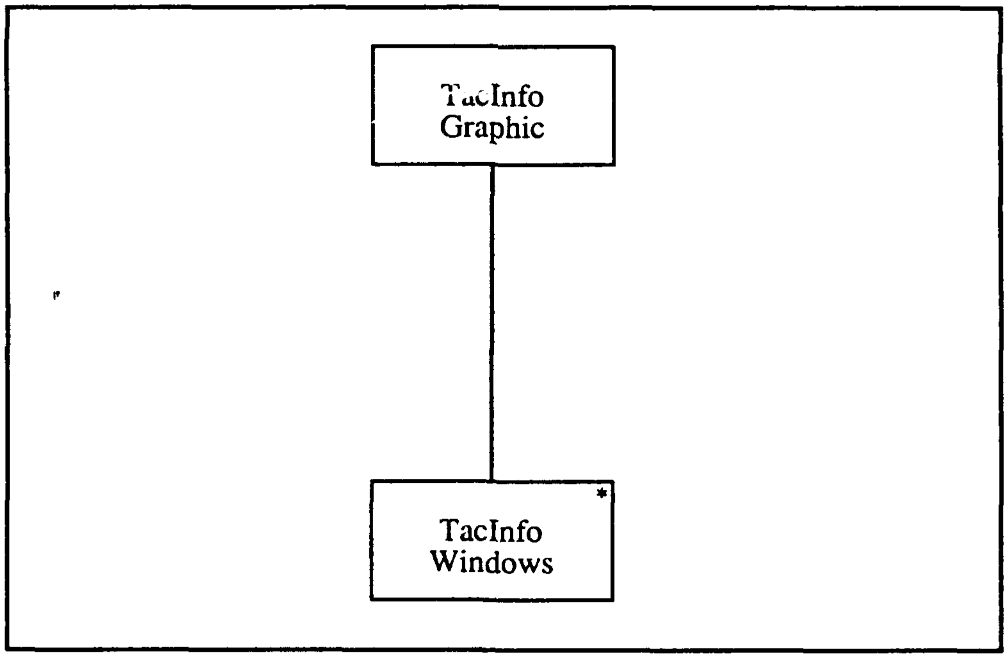


Figure 63: Data Structure Diagram - TacInfo_Graphic

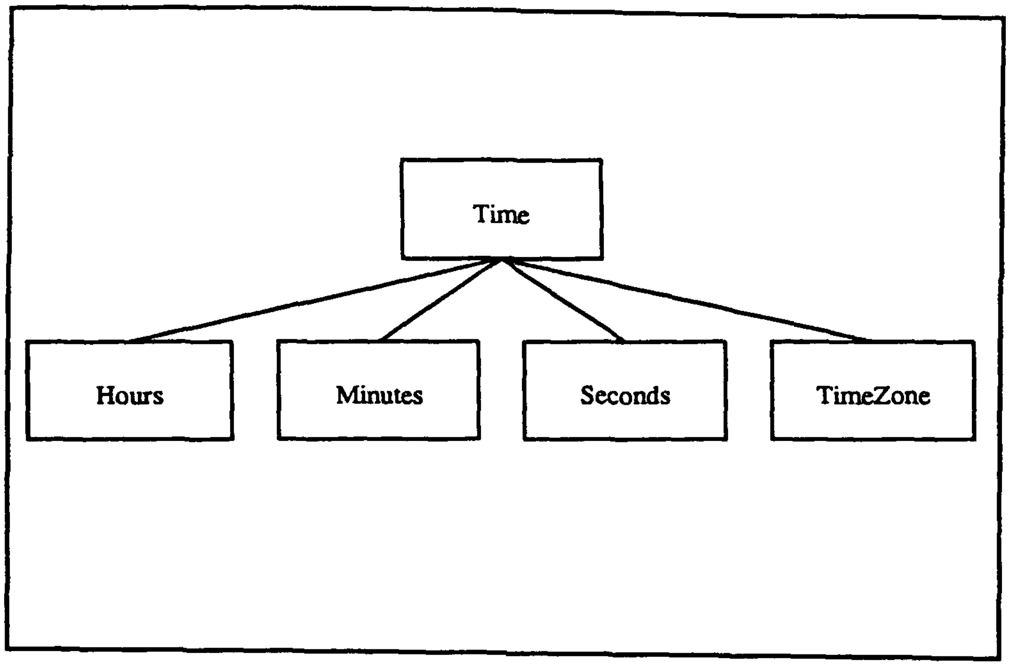


Figure 64: Data Structure Diagram - Time

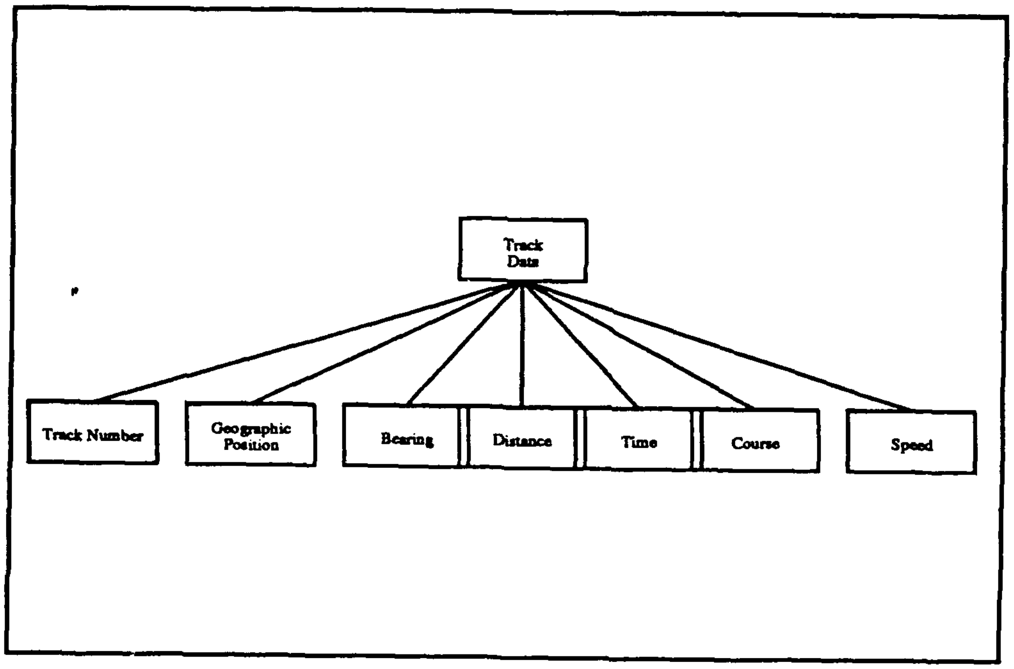


Figure 65: Data Structure Diagram - Track_Data

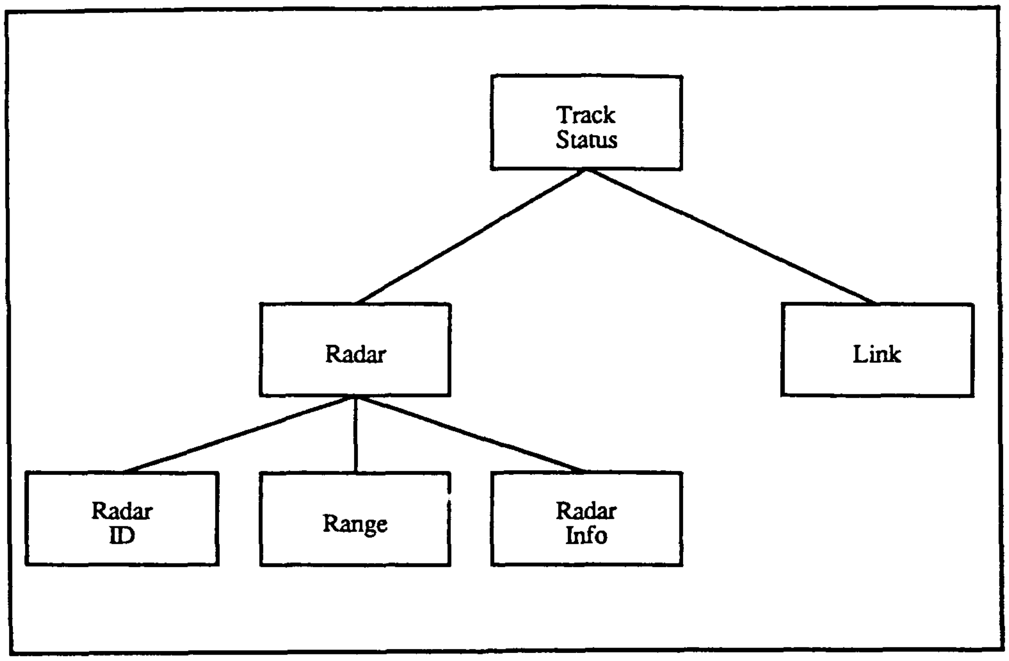


Figure 66: Data Structure Diagram - Track_Status

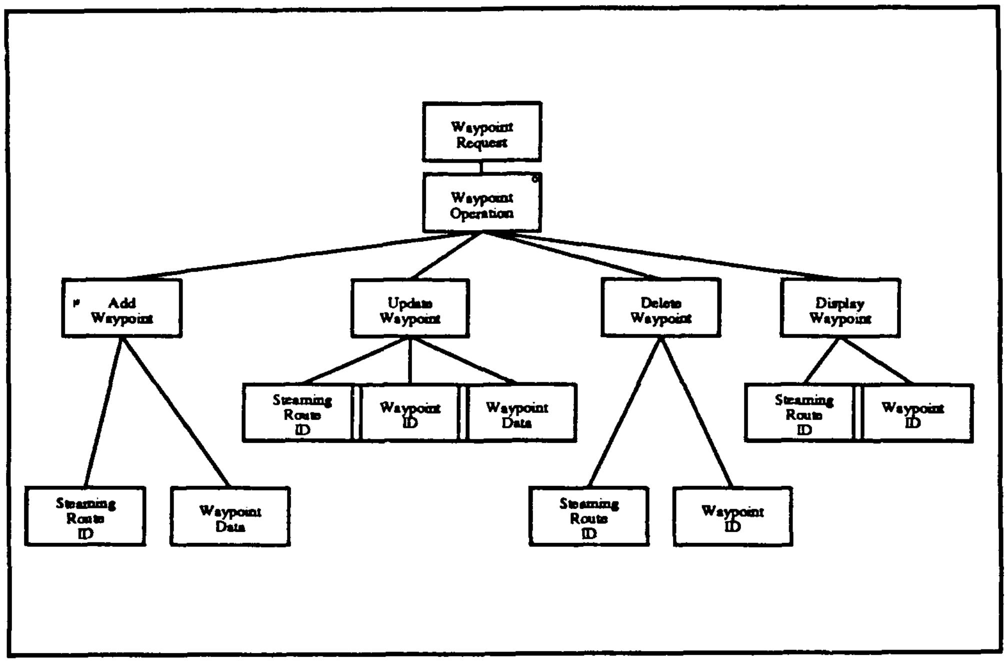


Figure 67: Data Structure Diagram - Waypoint_Request

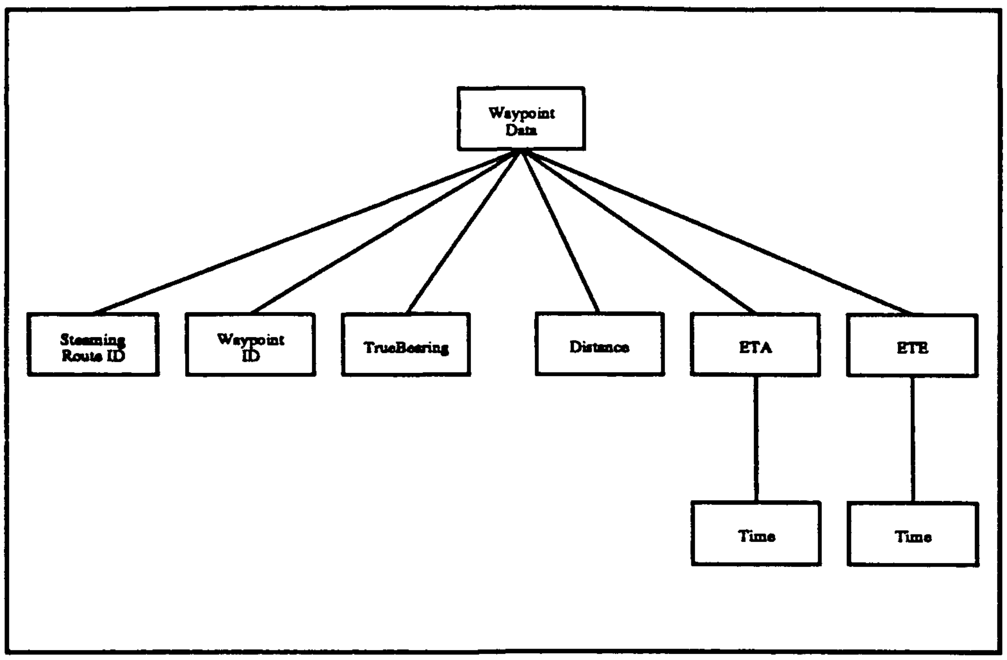


Figure 68: Data Structure Diagram - Waypoint_Data

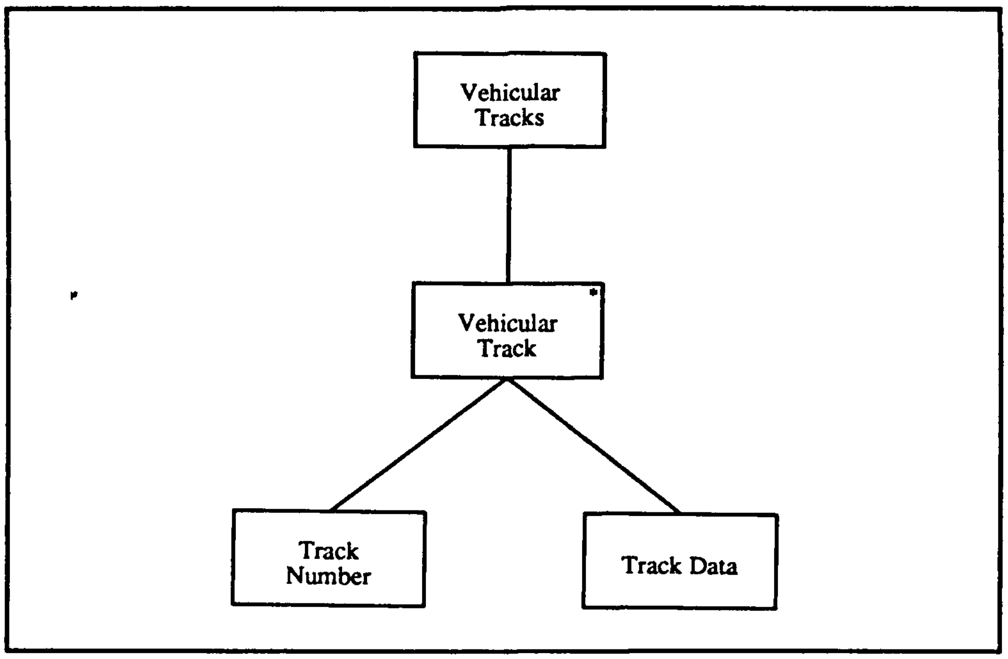


Figure 69: Data Structure Diagram - Vehicular Tracks

Further definition and low level information regarding data types and units of measure are provided in the next section as part of the Data Dictionary. Each individual block representing a data element in the data structure diagrams is fully defined in the Data Dictionary.

3. Data Dictionary

The Data Dictionary is probably the most crucial element of the Behavioral model. It is an organized listing of all the data elements that are pertinent to the system, with precise, rigorous definitions so that both user and systems analyst will have a common understanding of all inputs, outputs, components of data stores, and intermediate calculations [Ref. 18:p. 189]. The data dictionary explains the meaning of the flows between processes, composition of the data objects and values and units used in their definition.

A full listing of all data objects, and their component parts, which appear in the Top and Zero DFDs are provided in alphabetical order:

Name: Add_Waypoint

Defined as Sequence in Data Structure Diagram 'Waypoint_Request'

2 Components:

Steaming_Route_ID	Data Element
Waypoint_Data	Sequence

Used in 1 Data Structure
Waypoint_Operation

Name: Additional_Special_Point_Info

Defined as Data Element in Data Structure Diagram 'Special_Points'

Used in 1 Data Structure
Special_Points

Note DataDefinition
Name: Additional_Special_Point_Info
Type: string
Text field for operator use.

Name: Additional_TrackInfo

Defined as Sequence in Data Structure Diagram 'Additional_TrackInfo'

3 Components:

TrackType	Data Element
Hull_No	Data Element
Name	Data Element

Used in 4 Data Structures
Manual_Tracks
LinkTrackInfo
Display_Track
Tracks

Name: Band

Defined as Data Element in Data Structure Diagram 'Radar_Info'

Used in 1 Data Structure
  Radar_Info

Note DataDefinition
  Constraint: A..Z
  Name: Band
  Type: character
  Indicates the band of the selected radar. Band indicator as specified by current EW policy. Preset A..Z.

Name: Bearing

Defined as Data Element in Data Structure Diagram 'Manual_Tracks'

Used in 2 Data Structures
  Manual_Tracks
  Track_Data

Note DataDefinition
  Constraint: 000.00..359.00
  Name: Bearing
  Type: real
  Unit: degrees

Name: CPA_Data

Defined as Sequence in Data Structure Diagram 'CPA_Data'

4 Components:

Origin_Unit	Data Element
Time_To_CPA	Data Element
TrueBearing	Data Element
CPA_Distance	Data Element

Used in 1 Data Flow

DFD Level	From/To	
0	Process	3 Navigation_Handler
	Process	1 MMI

Name: CPA_Distance

Defined as Data Element in Data Structure Diagram 'CPA_Data'

Used in 1 Data Structure
CPA_Data

Note DataDefinition
Constraint: 00000..99999
Name: CPA_Distance
Type: integer
Units: yards

Name: CPA_Other_Track

Defined as Sequence in Data Structure Diagram 'CPA_Request'

2 Components:

Origin_Track	Sequence
Target_Track	Sequence

Used in 1 Data Structure
CPA_Selector

Note DataDefinition
Name: CPA_Other_Track
CPA Other Track requests CPA information on a selected Origin track (other than ownship) relative to a second selected Target track.

Name: CPA_Ownship

Defined as Sequence in Data Structure Diagram 'CPA_Request'

1 Component:

Track_Data	Sequence

Used in 1 Data Structure
CPA_Selector

Note DataDefinition
Name: CPA_Ownship
CPA Ownship requests CPA information on a selected track relative to ownship position, course and speed.

Name: CPA_Request

Defined as Selection in Data Structure Diagram 'CPA_Request'

1 Component:

CPA_Selector	Sequence

Used in 1 Data Flow

DFD Level	From/To	
0	Process	1 MMI
	Process	3 Navigation_Handler

Name: CPA_Selector

Defined as Sequence in Data Structure Diagram 'CPA_Request'

2 Components:

CPA_Ownship	Sequence
CPA_Other_Track	Sequence

Used in 1 Data Structure

CPA_Request

Name: Category

Defined as Data Element in Data Structure Diagram 'LinkTrackInfo'

Used in 3 Data Structures

Manual_Tracks
LinkTrackInfo
Display_Track

Note DataDefinition

Constraint: Air | Surface | Subsurface
Name: Category
Type: string

Name: Command

Used in 1 Data Structure
Operator_Commands

Note DataDefinition
Name: Command
Commands are generated by the "hook" function or by menu selection, in any order. Hooking a point on any active TacPlot shall cause the system to respond depending on what, if anything, was hooked. This may include prompts and menu actions. Making a menu selection shall cause the system to respond in some way as well, including prompting the user to hook an appropriate point for further program action.

Name: ContactID

Defined as Data Element in Data Structure Diagram 'Radar_Tracks'

Used in 1 Data Structure
Radar_Track

Note DataDefinition
Constraint: 0000..9999
Name: ContactID
Type: integer

Name: Course

Defined as Data Element in Data Structure Diagram 'Navigational_Messages'

Used in 5 Data Structures
Manual_Tracks
Ownship
Navigational_Message
LinkTrackInfo
Track_Data

Note DataDefinition
Constraint: 000.00..359.00
Name: Course
Type: real
Unit : degrees

Name: DLRP

Defined as Sequence in Data Structure Diagram 'System_InitData'

1 Component:

Geographic_Position	Sequence

Used in 1 Data Structure

System_InitData

Name: Delete_Waypoint

Defined as Sequence in Data Structure Diagram 'Waypoint_Request'

2 Components:

Steaming_Route_ID	Data Element
Waypoint_ID	Data Element

Used in 1 Data Structure

Waypoint_Operation

Name: Display_Track

Defined as Sequence in Data Structure Diagram 'Display_Tracks'

4 Components:

Track_Data	Sequence
Category	Data Element
Identity	Data Element
Additional_TrackInfo	Sequence

Used in 1 Data Structure

Display_Tracks

Name: Display_Tracks

Defined as Iteration in Data Structure Diagram 'Display_Tracks'

1 Component:

Display_Track	Sequence

Used in 1 Data Flow

DFD Level	From/To	
0	Process	6 Tactical_DBMS
	Process	1 MMI

Note DataDefinition
Name: Display_Tracks
Stream of tracks which qualify for output.

Name: Display_Waypoint

Defined as Sequence in Data Structure Diagram 'Waypoint_Request'

2 Components:

Steaming_Route_ID	Data Element
Waypoint_ID	Data Element

Used in 1 Data Structure
Waypoint_Operation

Name: Distance

Defined as Data Element in Data Structure Diagram 'Manual_Tracks'

Used in 3 Data Structures
Manual_Tracks
Waypoint_Data
Track_Data

Note DataDefinition
Constraint: 000.0..999.9
Name: Distance
Type: real
Unit: nautical miles

Name: ETA

Defined as Sequence in Data Structure Diagram 'Waypoint_Data'

1 Component:

Time	Sequence

Used in 1 Data Structure

Waypoint_Data

Name: ETE

Defined as Sequence in Data Structure Diagram 'Waypoint_Data'

1 Component:

Time	Sequence

Used in 1 Data Structure

Waypoint_Data

Name: Fault_ID

Defined as Data Element in Data Structure Diagram 'Radar_Info'

Used in 1 Data Structure

Faults

Note DataDefinition

Constraint: 00..99

Name: Fault_ID

Type: integer

Name: Faults

Defined as Iteration in Data Structure Diagram 'Radar_Info'

1 Component:

Fault_ID	Data Element

Used in 1 Data Structure

Radar_Info

Note DataDefinition

Name: Faults

Indicates faults in the selected radar. Complete when radar interface is specified.

Name: Geographic_Position

Defined as Sequence in Data Structure Diagram 'Geographic_Position'

2 Components:

Latitude	Sequence
Longitude	Sequence

Used in 10 Data Structures

Map_Positions
Map_Request
Manual_Tracks
Ownship
Navigational_Message
LinkTrackInfo
Link11_Track
Track_Data
DLRP
Ownship_Position

Name: Hours

Defined as Data Element in Data Structure Diagram 'Time'

Used in 1 Data Structure

Time

Note DataDefinition

Constraint: 00..23
Name: Hours
Type: integer

Name: Hull_No

Defined as Data Element in Data Structure Diagram 'Additional_TrackInfo'

Used in 1 Data Structure

Additional_TrackInfo

Note DataDefinition

Name: Hull_No
Type: string

Name: Identity

Defined as Data Element in Data Structure Diagram 'LinkTrackInfo'

Used in 3 Data Structures
- Manual_Tracks
- LinkTrackInfo
- Display_Track

Note DataDefinition
- Constraint: Friendly | Hostile | Unknown
- Name: Identity
- Type: string

Name: Julian_Date

Defined as Data Element in Data Structure Diagram 'System_InitData'

Used in 1 Data Structure
- System_InitData

Note DataDefinition
- Constraint: 0001..9365
- Name: Julian_Date
- Type: integer
- Julian date is simply the last digit of the current year followed by the day represented as a three digit integer which corresponds to a sequential count beginning on 1 January.

Name: LCCDSWS

Defined as Process in Data Flow Diagram 'top'

Process index: 0

6 Incoming Parameters

Link11_Tracks	Data Parameter
Map_Messages	Data Parameter
Navigational_Messages	Data Parameter
Operator_Commands	Data Parameter
Operator_Messages	Data Parameter
Radar_Tracks	Data Parameter

4 Outgoing Parameters

Load_Map	Data Parameter
System_Messages	Data Parameter
TacInfo_Alpha	Data Parameter
TacInfo_Graphic	Data Parameter

Name: LatDec

Defined as Data Element in Data Structure Diagram 'Geographic_Position'

Used in 1 Data Structure
Latitude

Note DataDefinition
Constraint: N | S
Name: LatDec
Type: character

Name: LatDegrees

Defined as Data Element in Data Structure Diagram 'Geographic_Position'

Used in 2 Data Structures

Load_Map
Latitude

Note DataDefinition

Constraint: 00..90
Name: LatDegrees
Type: integer
Unit: degrees

Name: Latitude

Defined as Sequence in Data Structure Diagram 'Geographic_Position'

4 Components:

LatDegrees	Data Element
Minutes	Data Element
Seconds	Data Element
LatDec	Data Element

Used in 1 Data Structure

Geographic_Position

Name: Link

Defined as Sequence in Data Structure Diagram 'Link'

2 Components:

Link11_On/Off	Data Element
Link11_Faults	Data Element

Used in 1 Data Structure

Track_Status

Name: Link11_Enable

Defined as Sequence in Data Structure Diagram 'Link11_Enable'

1 Component:

Link11_On/Off	Data Element

Used in 1 Data Flow

DFD Level	From/To	
0	Process	1 MMI
	Process	2 Track_Handler

Name: Link11_Faults

Defined as Data Element in Data Structure Diagram 'Link'

Used in 1 Data Structure
Link

Note DataDefinition
Constraint: 00..99
Name: Link11_Faults
Type: integer
Used for later implementation of link fault indication.

Name: Link11_On/Off

Defined as Data Element in Data Structure Diagram 'Link11_Enable'

Used in 2 Data Structures
Link11_Enable
Link

Note DataDefinition
Name: Link11_On/Off
Type: boolean
Activates task in track handler to accept Link11 tracks for processing.

Name: Link11_System

Defined as External in Data Flow Diagram 'top'

External Id: d

Name: Link11_Track

Defined as Sequence in Data Structure Diagram 'Link11_Tracks'

4 Components:

Time	Sequence
Geographic_Position	Sequence
Participating_Unit	Data Element
LinkTrackInfo	Sequence

Used in 1 Data Structure
  Link11_Tracks

Name: Link11_Tracks

Defined as Iteration in Data Structure Diagram 'Link11_Tracks'

1 Component:

Link11_Track	Sequence

Used in 2 Data Flows

DFD Level	From/To	
top	External	(d) Link11_System
	Process	0 LCCDSWS
0	Offpage Level 0	
	Process	2 Track_Handler

Name: LinkTrackInfo

Defined as Sequence in Data Structure Diagram 'LinkTrackInfo'

7 Components:

Category	Data Element
Identity	Data Element
Course	Data Element
Speed	Data Element
Time	Sequence
Geographic_Position	Sequence
Additional_TrackInfo	Sequence

Used in 1 Data Structure

Link11_Track

Name: Load_Map

Defined as Sequence in Data Structure Diagram 'Load_Map'

2 Components:

LatDegrees	Data Element
LongDegrees	Data Element

Used in 2 Data Flows

DFD Level	From/To	
top	Process	0 LCCDSWS
	External	(c) Shoreline_Database
0	Process	4 Map_Handler
	Offpage Level 0	

Note DataDefinition

Name: Load_Map

The Load_Map datastructure is defined in terms of DMA map data-management. Each map chunk corresponds to one degree (60min Lat by 60min Long) of the earth surface.

Name: LongDec

Defined as Data Element in Data Structure Diagram 'Geographic_Position'

Used in 1 Data Structure
  Longitude

Note DataDefinition
  Constraint: E | W
  Name: LongDec
  Type: character

Name: LongDegrees

Defined as Data Element in Data Structure Diagram 'Geographic_Position'

Used in 2 Data Structures
  Load_Map
  Longitude

Note DataDefinition
  Constraint: 000..179
  Name: LongDegrees
  Type: integer
  Unit: degrees

Name: Longitude

Defined as Sequence in Data Structure Diagram 'Geographic_Position'

4 Components:

LongDegrees	Data Element
Minutes	Data Element
Seconds	Data Element
LongDec	Data Element

Used in 1 Data Structure
  Geographic_Position

Note DataDefinition
  Name: Longitude

Name: MMI

Defined as Process in Data Flow Diagram '0'

Process index: 1

8 Incoming Parameters

CPA_Data	Data Parameter
Display_Tracks	Data Parameter
Map_Data	Data Parameter
Navigation_Data	Data Parameter
Operator_Commands	Data Parameter
Operator_Messages	Data Parameter
Track_Status	Data Parameter
Waypoint_Data	Data Parameter

14 Outgoing Parameters

CPA_Request	Data Parameter
Link11_Enable	Data Parameter
Manual_Tracks	Data Parameter
Map_Request	Data Parameter
NavStatus	Data Parameter
Nav_Enable	Data Parameter
Radar_Enable	Data Parameter
Range_Select	Data Parameter
Special_Points	Data Parameter
System_InitData	Data Parameter
System_Messages	Data Parameter
TacInfo_Alpha	Data Parameter
TacInfo_Graphic	Data Parameter
Waypoint_Request	Data Parameter

Name: Manual_Tracks

Defined as Sequence in Data Structure Diagram 'Manual_Tracks'

9 Components:

Category	Data Element
Identity	Data Element
Bearing	Data Element
Distance	Data Element
Geographic_Position	Sequence
Course	Data Element
Speed	Data Element
Time	Sequence
Additional_TrackInfo	Sequence

Used in 1 Data Flow

DFD Level	From/To	
0	Process	1 MMI
	Process	2 Track_Handler

Name: Map_Data

Defined as Sequence in Data Structure Diagram 'Map_Data'

2 Components:

Map_Number	Data Element
Map_Positions	Iteration

Used in 1 Data Flow

DFD Level	From/To	
0	Process	4 Map_Handler
	Process	1 MMI

Name: Map_Handler

Defined as Process in Data Flow Diagram '0'

Process index: 4

2 Incoming Parameters

Map_Messages	Data Parameter
Map_Request	Data Parameter

2 Outgoing Parameters

Load_Map	Data Parameter
Map_Data	Data Parameter

Name: Map_Messages

Defined as Sequence in Data Structure Diagram 'Map_Messages'

2 Components:

Map_Number	Data Element
Map_Positions	Iteration

Used in 2 Data Flows

DFD Level	From/To	
top	External	(c) Shoreline_Database
	Process	0 LCCDSWS
0	Offpage Level 0	
	Process	4 Map_Handler

Name: Map_Number

Defined as Data Element in Data Structure Diagram 'Map_Messages'

Used in 2 Data Structures

Map_Messages

Map_Data

Note DataDefinition

Constraint: 0000..9999

Name: Map_Number

Type: integer

Name: Map_Positions

Defined as Iteration in Data Structure Diagram 'Map_Messages'

1 Component:

Geographic_Position	Sequence

Used in 2 Data St.uctures

Map_Messages
Map_Data

Name: Map_Request

Defined as Sequence in Data Structure Diagram 'Map_Request'

1 Component:

Geographic_Position	Sequence

Used in 1 Data Flow

DFD Level	From/To	
0	Process	1 MMI
	Process	4 Map_Handler

Name: Message

Note DataDefinition

Name: Message

A message consists of data the system must obtain
from the operator.

The system shall prompt the user for required information based on currently active commands and processes. The information shall be provided via menu selection process, data entry into system generated Forms and templates, etc.

Name: Minutes

Defined as Data Element in Data Structure Diagram 'Geographic_Position'

Used in 3 Data Structures
- Time
- Latitude
- Longitude

Note DataDefinition
- Constraint: 00..59
- Name: Minutes
- Type: integer
- Unit: minutes

Name: Mode

Defined as Data Element in Data Structure Diagram 'Radar_Info'

Used in 1 Data Structure
- Radar_Info

Note DataDefinition
- Constraint: MTI|Non-MTI|Surface
- Name: Mode
- Type: string
- The indicated modes are assumptions and have to be adapted when radar interface is specified.

Name: Name

Defined as Data Element in Data Structure Diagram 'Additional_TrackInfo'

Used in 1 Data Structure
- Additional_TrackInfo

Note DataDefinition
- Name: Name
- Type: string

Name: NavStatus

Defined as Data Element in Data Structure Diagram 'NavStatus'

Used in 2 Data Structures
Navigational_Message
Navigation_Data

Used in 1 Data Flow

DFD Level	From/To	
0	Process	1 MMI
	Process	6 Tactical_DBMS

Note DataDefinition
Name: NavStatus
Type: boolean
At this time NavStatus does not contain information about the subsystems of the Navigational Systems. These have to be defined at a later time and will need a fault indicator for LCCDSWS.

Name: Nav_Enable

Defined as Data Element in Data Structure Diagram 'Nav_Enable'
Used in 1 Data Flow

DFD Level	From/To	
0	Process	1 MMI
	Process	3 Navigation_Handler

Note DataDefinition
Name: Nav_Enable
Type: boolean
True = Navigational Messages are accepted by the Navigation Handler.
False = Navigational Messages are ignored by the Navigation Handler. Ownship no longer updated from external source.

Name: Navigation_Data

Defined as Sequence in Data Structure Diagram 'Navigation_Data'

2 Components:

Ownship	Sequence
NavStatus	Data Element

Used in 1 Data Flow

DFD Level	From/To	
0	Process	3 Navigation_Handler
	Process	1 MMI

Name: Navigation_Handler

Defined as Process in Data Flow Diagram '0'

Process index: 3

5 Incoming Parameters

CPA_Request	Data Parameter
Nav_Enable	Data Parameter
Navigational_Messages	Data Parameter
System_InitData	Data Parameter
Waypoint_Request	Data Parameter

4 Outgoing Parameters

CPA_Data	Data Parameter
Navigation_Data	Data Parameter
Ownship	Data Parameter
Waypoint_Data	Data Parameter

Name: Navigation_System

Defined as External in Data Flow Diagram 'top'
External Id: a

Name: Navigational_Message

Defined as Sequence in Data Structure Diagram 'Navigational_Messages'

5 Components:

Geographic_Position	Sequence
Course	Data Element
Speed	Data Element
Time	Sequence
NavStatus	Data Element

Used in 1 Data Structure
Navigational_Messages

Note DataDefinition
Name: Navigational_Message

Name: Navigational_Messages

Defined as Iteration in Data Structure Diagram 'Navigational_Messages'

1 Component:

Navigational_Message	Sequence

Used in 2 Data Flows

DFD Level	From/To	
top	External	(a) Navigation_System
	Process	0 LCCDSWS
0	Offpage Level 0	
	Process	3 Navigation_Handler

Name: Offpage.0

Defined as Offpage Connector in Data Flow Diagram '0'

Name: Offpage.top

Defined as Offpage Connector in Data Flow Diagram 'top'

Name: Operator

Defined as External in Data Flow Diagram 'top'

External Id: e

Name: Operator_Commands

Defined as Iteration in Data Structure Diagram 'Operator_Commands'

1 Component:
    Command

Used in 2 Data Flows

```
DFD Level  From/To
      top  External           (e)  Operator
           Process             0  LCCDSWS
        0  Offpage Level 0
           Process             1  MMI
```

Name: Operator_Message

Defined as Undefined Data

Used in 1 Data Structure
    Operator_Messages

Note DataDefinition
    Name: Operator_Message
    A message consists of data the system must obtain from the operator. The system shall prompt the user for required information based on currently active commands and processes. The information shall be provided via menu selection process, data entry into system generated forms and templates, etc.

Name: Operator_Messages

Defined as Iteration in Data Structure Diagram 'Operator_Messages'

1 Component:

Operator_Message	Undefined Data

Used in 2 Data Flows

DFD Level	From/To	
top	External	(e) Operator
	Process	0 LCCDSWS
0	Offpage Level 0	
	Process	1 MMI

Name: Origin

Defined as Data Element in Data Structure Diagram 'System_Tracks'

Used in 1 Data Structure
Tracks

Note DataDefinition
Constraint: Local Manual I Local Auto I Remote
Name: Origin
Type: string
Indicates source of track data:
Local Manual: Operator,
Local Auto: Radar,
Remote: Link11.

Name: Origin_Track

Defined as Sequence in Data Structure Diagram 'CPA_Request'

1 Component:

Track_Data	Sequence

Used in 1 Data Structure
CPA_Other_Track

Name: Origin_Unit

Defined as Data Element in Data Structure Diagram 'CPA_Data'

Used in 1 Data Structure
CPA_Data

Note DataDefinition
Constraint: OwnUnit | OriginTrack
Name: Origin_Unit
Type: string

Name: Ownship

Defined as Sequence in Data Structure Diagram 'Ownship'

4 Components:

Geographic_Position	Sequence
Course	Data Element
Speed	Data Element
Time	Sequence

Used in 1 Data Structure
Navigation_Data

Used in 1 Data Flow

DFD Level	From/To	
0	Process	3 Navigation_Handler
	Process	2 Track_Handler

Note DataDefinition
Name: Ownship

Name: Ownship_Position

Defined as Sequence in Data Structure Diagram 'System_InitData'

1 Component:

Geographic_Position	Sequence

Used in 1 Data Structure
System_InitData

Name: Participating_Unit

Defined as Data Element in Data Structure Diagram 'Link11_Tracks'

Used in 1 Data Structure
Link11_Track

Note DataDefinition
Constraint: 00..76
Name: Participating_Unit
Type: integer
Octal Number representing Link 11 participating units.

Name: Radar

Defined as Sequence in Data Structure Diagram 'Track_Status'

3 Components:

Radar_ID	Data Element
Range	Data Element
Radar_Info	Sequence

Used in 1 Data Structure
Track_Status

Name: Radar_Enable

Defined as Sequence in Data Structure Diagram 'Radar_Enable'

2 Components:

Radar_ID	Data Element
Radar_On/Off	Data Element

Used in 1 Data Flow

DFD Level	From/To	
0	Process	1 MMI
	Process	2 Track_Handler

Note DataDefinition
Name: Radar_Enable
Activates tasks in track handler to accept tasks for processing of tracks from the selected radar identified by Radar_ID. Radar_ID shall set to the range of radars available, preset 0..9.

Name: Radar_ID

Defined as Data Element in Data Structure Diagram 'Radar_Enable'

Used in 2 Data Structures
Radar_Enable
Radar

Note DataDefinition
Constraint: 0..9
Name: Radar_ID
Type: integer
Identifies the selected radar.

Name: Radar_Info

Defined as Sequence in Data Structure Diagram 'Radar_Info'

4 Components:

Mode	Data Element
Standby	Data Element
Faults	Iteration
Band	Data Element

Used in 1 Data Structure
Radar

Note DataDefinition
Name: Radar_Info
Radar_Info will held additional informations about the status of the selected radar e.g., mode, faults, band, transmitter on/off.
Has to be determined completely when radar interface is fully specified.

Name: Radar_On/Off

Defined as Data Element in Data Structure Diagram 'Radar_Enable'

Used in 1 Data Structure
Radar_Enable

Note DataDefinition
Name: Radar_On/Off
Type: boolean
true = radar on
false = radar off.

Name: Radar_System

Defined as External in Data Flow Diagram 'top'

External Id: b

Name: Radar_Track

Defined as Sequence in Data Structure Diagram 'Radar_Tracks'

3 Components:

Range	Data Element
TrueBearing	Data Element
ContactID	Data Element

Used in 1 Data Structure
Radar_Tracks

Name: Radar_Tracks

Defined as Iteration in Data Structure Diagram 'Radar_Tracks'

1 Component:

Radar_Track	Sequence

Used in 2 Data Flows

DFD Level	From/To	
top	External	(b) Radar_System
	Process	0 LCCDSWS
0	Offpage Level 0	
	Process	2 Track_Handler

Name: Range

Defined as Data Element in Data Structure Diagram 'Radar_Tracks'

Used in 2 Data Structures

Radar_Track

Radar

Note DataDefinition

Constraint: 000.0..999.9

Name: Range

Type: real

Unit : Nautical Miles

Name: Range_Select

Defined as Data Element in Data Structure Diagram 'Range_Select'

Used in 1 Data Flow

DFD Level	From/To	
0	Process	1 MMI
	Process	6 Tactical_DBMS

Note DataDefinition

Constraint: 2|4|8|16|32|64|128|256|512

Name: Range_Select

Type: integer

Name: Seconds

Defined as Data Element in Data Structure Diagram 'Geographic_Position'

Used in 3 Data Structures

Time

Latitude

Longitude

Note DataDefinition

Constraint: 00..59

Name: Seconds

Type: integer

Unit: seconds

Name: Shoreline_Database

Defined as External in Data Flow Diagram 'top'

External Id: c

Name: Special_Point_Category

Defined as Data Element in Data Structure Diagram 'Special_Points'

Used in 1 Data Structure
Special_Points

Note DataDefinition
Constraint: Man in Water | Formation Center | PIM | Nav Hazard | DLRP | Reference Point
Name: Special_Point_Category
Type: string

Name: Special_Points

Defined as Sequence in Data Structure Diagram 'Special_Points'

3 Components:

Special_Point_Category	Data Element
Track_Data	Sequence
Additional_Special_Point_Info	Data Element

Used in 1 Data Flow

DFD Level	From/To	
0	Process	1 MMI
	Process	2 Track_Handler

Note DataDefinition
Name: Special_Points
All Special Points have certain geographic information which is passed via the Track Data object. In some cases not all the data contained in Track Data is applicable. For instance, only the Formation Center and PIM special points have course/speed info associated with them.

Name: Speed

Defined as Data Element in Data Structure Diagram 'Navigational_Messages'

Used in 5 Data Structures

Manual_Tracks
Ownship
Navigational_Messages
LinkTrackInfo
Track_Data

Note DataDefinition

Constraint: 000.0..999.9
Name: Speed
Type: real
Unit : knots

Name: Standby

Defined as Data Element in Data Structure Diagram 'Radar_Info'

Used in 1 Data Structure

Radar_Info

Note DataDefinition

Name: Standby
Type: boolean
Indicates whether radar is transmitting (false) or in standby (true).

Name: Steaming_Route_ID

Defined as Data Element in Data Structure Diagram 'Waypoint_Data'

Used in 5 Data Structures

Waypoint_Data
Add_Waypoint
Update_Waypoint
Delete_Waypoint
Display_Waypoint

Note DataDefinition

Constraint: 1..6
Name: Steaming_Route_ID
Type: integer

Name: System_InitData

Defined as Sequence in Data Structure Diagram 'System_InitData'

4 Components:

DLRP	Sequence
Ownship_Position	Sequence
Julian_Date	Data Element
Time	Sequence

Used in 1 Data Flow

DFD Level	From/To	
0	Process	1 MMI
	Process	3 Navigation_Handler

Name: System_Message

Used in 1 Data Structure

System_Messages

Note DataDefinition

Name: System_Message

Messages shall consist of structures defining the information provided to the user by the system including prompts, cues, alerts, warnings, help, etc. The system shall generate these messages independently or as a result of operator actions/commands.

Name: System_Messages

Defined as Iteration in Data Structure Diagram 'System_Messages'

1 Component:

System_Message	Undefined Data

Used in 2 Data Flows

DFD Level	From/To	
top	Process	0 LCCDSWS
	External	(e) Operator
0	Process	1 MMI
	Offpage Level 0	

Name: System_Tracks

Defined as Iteration in Data Structure Diagram 'System_Tracks'

1 Component:

Tracks	Sequence

Used in 1 Data Flow

DFD Level	From/To	
0	Process	2 Track_Handler
	Process	6 Tactical_DBMS

Name: TacInfo_Alpha

Defined as Iteration in Data Structure Diagram 'TacInfo_Alpha'

1 Component:

TacInfo_Tableau	Undefined Data

Used in 2 Data Flows

DFD Level	From/To	
top	Process	0 LCCDSWS
	External	(e) Operator
0	Process	1 MMI
	Offpage Level 0	

Name: TacInfo_Graphic

Defined as Iteration in Data Structure Diagram 'TacInfo_Graphic'

1 Component:

TacInfo_Windows	Undefined Data

Used in 2 Data Flows

DFD Level	From/To	
top	Process	0 LCCDSWS
	External	(e) Operator
0	Process	1 MMI
	Offpage Level 0	

Name: TacInfo_Tableau

Defined as Undefined Data

Used in 1 Data Structure
TacInfo_Alpha

Note DataDefinition
Name: TacInfo_Tableau
TacInfo Tableaux are individual structures containing a set of related information which must be accessible to the operator in alphanumeric form. These tableaux shall support output of tactical data and information as well as user data input. The system shall continuously update the tableaux as information they display is changed.

Name: TacInfo_Windows

Defined as Undefined Data

Used in 1 Data Structure
TacInfo_Graphic

Note DataDefinition
Name: TacInfo_Windows
TacInfo Windows provide the method for organization of the system and tactical information presentation to the user. Graphics and graphical data are displayed within the structure of a window environment. This environment defines the Tactical Display. The basic ingredients of the Tactical Display are; TacPlot windows, a system of menu windows and Tableau windows.

Name: Tactical_DBMS

Defined as Process in Data Flow Diagram '0'

Process index: 6

4 Incoming Parameters

NavStatus	Data Parameter
Range_Select	Data Parameter
System_Tracks	Data Parameter
Vehicular_Tracks	Data Parameter

2 Outgoing Parameters

Display_Tracks	Data Parameter
Vehicular_Tracks	Data Parameter

Name: Target_Track

Defined as Sequence in Data Structure Diagram 'CPA_Request'

1 Component:

Track_Data	Sequence

Used in 1 Data Structure
CPA_Other_Track

Name: Time

Defined as Sequence in Data Structure Diagram 'Time'

4 Components:

Hours	Data Element
Minutes	Data Element
Seconds	Data Element
TimeZone	Data Element

Used in 9 Data Structures

- Manual_Tracks
- Ownship
- Navigational_Messages
- LinkTrackInfo
- Link11_Track
- Track_Data
- ETA
- ETE
- System_InitData

Note DataDefinition
Name: Time

Name: TimeZone

Defined as Data Element in Data Structure Diagram 'Time'

Used in 1 Data Structure

- Time

Note DataDefinition

Constraint: A..Z
Name: TimeZone
Type: character
Default value should be 'Z' for "zulu" or Universal Coordinated Time.

Name: Time_To_CPA

Defined as Data Element in Data Structure Diagram 'CPA_Data'

Used in 1 Data Structure
CPA_Data

Note DataDefinition
Constraint: 000..999
Name: Time_To_CPA
Type: integer
Unit: minutes

Name: TrackType

Defined as Data Element in Data Structure Diagram 'Additional_TrackInfo'

Used in 1 Data Structure
Additional_TrackInfo

Note DataDefinition
Name: TrackType
Type: string

Name: Track_Data

Defined as Sequence in Data Structure Diagram 'Track_Data'

7 Components:

Track_Number	Data Element
Geographic_Position	Sequence
Bearing	Data Element
Distance	Data Element
Time	Sequence
Course	Data Element
Speed	Data Element

Used in 7 Data Structures

- Special_Points
- Display_Track
- Vehicular_Track
- CPA_Ownship
- Origin_Track
- Target_Track
- Tracks

Name: Track_Handler

Defined as Process in Data Flow Diagram '0'

Process index: 2

7 Incoming Parameters

Link11_Enable	Data Parameter
Link11_Tracks	Data Parameter
Manual_Tracks	Data Parameter
Ownship	Data Parameter
Radar_Enable	Data Parameter
Radar_Tracks	Data Parameter
Special_Points	Data Parameter

2 Outgoing Parameters

System_Tracks	Data Parameter
Track_Status	Data Parameter

Name: Track_Info

Used in 1 Data Structure
Tracks

Name: Track_Number

Defined as Data Element in Data Structure Diagram 'Track_Data'

Used in 1 Data Structure
Track_Data

Note DataDefinition
Constraint: 0000..7776
Name: Track_Number
Type: integer

Name: Track_Status

Defined as Sequence in Data Structure Diagram 'Track_Status'

2 Components:

Radar	Sequence
Link	Sequence

Used in 1 Data Flow

DFD Level	From/To	
0	Process	2 Track_Handler
	Process	1 MMI

Name: Tracking_Filter

Defined as Process in Data Flow Diagram '0'

Process index: 5

1 Incoming Parameter

Vehicular_Tracks	Data Parameter

1 Outgoing Parameter

Vehicular_Tracks	Data Parameter

Name: Tracks

Defined as Sequence in Data Structure Diagram 'System_Tracks'

3 Components:

Track_Data	Sequence
Additional_TrackInfo	Sequence
Origin	Data Element

Used in 1 Data Structure
System_Tracks

Name: TrueBearing

Defined as Data Element in Data Structure Diagram 'Radar_Tracks'

Used in 3 Data Structures
CPA_Data
Waypoint_Data
Radar_Track

Note DataDefinition
Constraint: 000.00..999.99
Name: TrueBearing
Type: real
Unit: degrees

Name: Update_Waypoint

Defined as Sequence in Data Structure Diagram 'Waypoint_Request'

3 Components:

Steaming_Route_ID	Data Element
Waypoint_ID	Data Element
Waypoint_Data	Sequence

Used in 1 Data Structure
Waypoint_Operation

Name: Vehicular_Track

Defined as Sequence in Data Structure Diagram 'Vehicular_Tracks'

1 Component:

Track_Data	Sequence

Used in 1 Data Structure
Vehicular_Tracks

Name: Vehicular_Tracks

Defined as Iteration in Data Structure Diagram 'Vehicular_Tracks'

1 Component:

Vehicular_Track	Sequence

Used in 2 Data Flows

DFD Level	From/To	
0	Process	5 Tracking_Filter
	Process	6 Tactical_DBMS
0	Process	6 Tactical_DBMS
	Process	5 Tracking_Filter

Name: Waypoint_Data

Defined as Sequence in Data Structure Diagram 'Waypoint_Data'

6 Components:

Steaming_Route_ID	Data Element
Waypoint_ID	Data Element
TrueBearing	Data Element
Distance	Data Element
ETA	Sequence
ETE	Sequence

Used in 2 Data Structures
Add_Waypoint
Update_Waypoint

Used in 1 Data Flow

DFD Level	From/To	
0	Process	3 Navigation_Handler
	Process	1 MMI

Name: Waypoint_ID

Defined as Data Element in Data Structure Diagram 'Waypoint_Data'

Used in 4 Data Structures
Waypoint_Data
Update_Waypoint
Delete_Waypoint
Display_Waypoint

Note DataDefinition
Constraint: 1..50
Name: Waypoint_ID
Type: integer

Name: Waypoint_Operation

Defined as Sequence in Data Structure Diagram 'Waypoint_Request'

4 Components:

Add_Waypoint	Sequence
Update_Waypoint	Sequence
Delete_Waypoint	Sequence
Display_Waypoint	Sequence

Used in 1 Data Structure
Waypoint_Request

Note DataDefinition
Name: Waypoint_Operation
Waypoint_Request is either a:
Add Waypoint or
-Update_Waypoint,
-Delete_Waypoint.

Name: Waypoint_Request

Defined as Selection in Data Structure Diagram 'Waypoint_Request'

1 Component:

Waypoint_Operation	Sequence

Used in 1 Data Flow

DFD Level	From/To	
0	Process	1 MMI
	Process	3 Navigation_Handler

IV. MAN-MACHINE INTERFACE DESIGN GUIDELINES

During the requirements analysis for the LCCDSWS it became obvious that the MMI would be one of the most important components of the proposed software system. Much research was conducted to determine the issues of an effective user interface in a tactical environment. We intend to provide an MMI for the LCCDS that is consistent with current Navy development efforts for a standardized workstation interface [Ref. 5]. The outcome of this research is included in Chapter II as Goal 1, the goal hierarchy for the MMI. However, it seems important to us not to lose the findings of the research carried out on user interface issues. For this reason we add this chapter to the LCCDSWS requirements analysis. It shall provide some of the underlying ideas for further design and implementation of the MMI as specified in Goal 1 and represents in some sense an environment model for a standardized MMI software system. Recommendations are further developed based on ideas and inputs from experienced fleet operators and Surface Warfare Officers along with technology being explored at the Naval Postgraduate School, the various Navy systems commands and laboratory facilities.

A. LCCDS USER CONCEPTUAL MODEL

The single most important ingredient of any interactive computer system is the user interface, the direct connection through which all interaction between man and machine takes place. This interface is often the critical factor in the success or failure of a system [Ref. 19:p. 1].

Before the design of a man-machine interface with even a modest chance for success can begin, the system designers must have a thorough understanding of the people who will use it. A *user model* must be developed. The user model is an attempt to define the characteristics and attributes associated with that user, or group of users, which will play a large part in the perceptions they form regarding the performance and functional utility of the system. The user interface must support a high level concept of the machine/system that the user can understand and relate directly to the tasks he must perform.

A user model for military systems can be fairly straight-forward to develop. Much of the model can be abstracted from pre-existing information on mission, tactical doctrine, terminology and training. A particular user can be roughly described as a novice, professional or expert as appropriate to his skills and abilities to handle the system. For our purposes it is sufficient to describe a user by the events that relate the real world environment to the corresponding tactical decisions, recommendations and ensuing actions he is responsible for:

- Recognition of the information
- Interpreting the meaning of that information
- Decision
- Reaction

With this in mind the following assumptions regarding operators of the LCCDS are made:

(1) There will be a wide spectrum of user experience and ability with NTDS and CDS systems ranging from the newest seaman to the senior petty officer and Chief petty officer levels.

(2) An experienced Watch Supervisor will be available to guide and assist the newly trained operators.

(3) All operators will have a basic understanding of the requirements of their particular rating to include the following:

- basic maneuvering board relationships
- navigation
- radar
- NTDS symbology and terminology
- Navy Standard Operating Procedures (SOP) for ships underway

In addition to the military training and experience aspects of the user group, the characteristics and capabilities of humans in general must be acknowledged and adapted to by the man-machine interface. Some of the important physiological and psychological factors to be considered are:

- human performance will degrade over time due to fatigue, boredom or attitude.
- visual detail is obtained over a narrow range (about 2 degrees) of eyesight [Ref. 20:p. 25].
- aural/audio communication can be processed by a human faster than any other method [Ref. 19:p. 422].
- effective capacity of human working (short term) memory is limited to a range of 5 to 9 different things ("chunks" of information) [Ref. 20:p. 39].

With an awareness of these important cognitive factors, the designer must create an interface which will reliably perform in a manner expected by the user. The users confidence in the system is directly related to how well the interface supports his conceptual model of what the system is and does. The users conceptual model of the LCCDS is expected to be something like the following:

> The LCCDS is a visual, map-like representation of a portion of the Earth upon which tactically significant tracks, representing real world objects, shall be imposed. Tracks shall move on the display with respect

to accurate time and spatial measurements obtained from the ships organic sensors (radar, navigational, visual, ESM, Link). The real world objects represented on the LCCDS display shall be associated with NTDS standard symbology and data.

Through the use of menus, cues, alerts and data tableaus the user shall be led through all actions involving control of the tactical display and be allowed to manipulate data and filter information in a manner consistent with mission requirements and procedures. The system shall provide reliable guides to cue the user for normal and emergency/critical response as well as keeping him informed about where he is in the current process and what the system status is.

B. FUNCTIONS OF A TACTICAL MAN-MACHINE INTERFACE

The typical MMI of current CDS's is the Standard Navy Display Console (SNDC). These consoles are usually daisy-chained in groups to a central computer and provide a variety of functions for interaction with the software system and control of subsystems. Functions like Range Select, Track Filtering and Action Entries, to name only a few, are implemented in discrete hardware provided by the consoles, making these devices complex, expensive and specialized for their particular purpose. To provide an operator with the basic skills for using a SNDC requires training periods measured in terms of months. For further information on SNDC see the Combat Direction Systems Specification for Surface Ships [Ref. 7].

During the requirements analysis for the LCCDS it became obvious that the development of a MMI would be one of the main issues for this project. The implementation of the LCCDS as a subset of existing CDS's makes it necessary to implement CDS functionality as provided by SNDC on bitmapped workstation displays that do not provide the hardware features of a SNDC.

The prime objective of the Man-Machine Interface is to insulate the user from the non-essential detail and highlight the information he needs to have. This must be done in a manner which provides intuitive tools for accurately representing the tactical problem comprehensibly to the user. Humans normally do not perform well in situations requiring concurrent thought processes. If the user must concentrate on operating the tool, he cannot also be concentrating on the information it provides him. The representation of tactical information using current workstation technology (supporting color, aural cues and speech recognition) can allow operators to make better, faster and more accurate tactical decisions, with significantly less training and experience.

Irregardless of mission or tactical circumstance, a ship's external sensors provide only one thing, raw data. This data becomes useful only after human analysis assigns meaning to it. The key to this crucial step lies in the capabilities and design of the MMI. The technology exists which will allow the MMI to be the "expert" at translating the real world environment into a manageable subset of tactically significant information, easily recognizable by the user.

A pre-defined, well-understood "Display Doctrine" (see Goal 1.3) provides the primitive guidelines by which the MMI filters, molds and represents tactical information in some form which attaches the desired priority and significance to the data it passes to the user. An important part of this MMI is its ability to extend over new and changing situations and requirements. It must be easy to modify and add new "filters" to the display doctrine. A ship may have many different missions or operations, therefore the MMI must be capable of providing data from many different tactical perspectives.

To give a simplified view on the process of collecting filtering and displaying tactical significant information consider Figure 70. A sensor, typically a radar, sonar or optical device detects environmental phenomena and converts it into quantifiable data. Problems with these first filters are that not all phenomena are recognized and not all recognized phenomena will finally turn out to be tactically significant. In the next step, the data is collected and stored in some way in a database system. Now the next filter can be applied on the stored data in order to get knowledge about the origin and nature of the phenomena associated with the data. Algorithms determine positions, course, speed and other characteristics of the phenomena, changing data into meaningful information. The information is further refined, via Display Doctrine, as it passes to the MMI. Information must be organized into a human understandable form. An optimization takes place, that allows the user to recognize the phenomena in the fastest possible way. The goal is to provide the user with only that information that is tactically significant or meaningful without any losses. At this point we can identify one of the critical functions of a tactical MMI:

(1) An effective man-machine-interaction requires the representation of environmental phenomena as meaningful information with respect to Human Information Processing.

The information has now arrived at its destination, the user. Clearly Interpretation and Decision are based on the cognitive skills and experience of the user. The domain of the MMI is the support of Recognition and Reaction. The issue on recognition is described in (1). The question is how to present information, without loss, to the user to ensure fastest possible recognition without reducing performance and attention.

A reaction may be a variety, or range, of responses including direct intervention into the environment (e.g. releasing a weapon), change of own behavior or change of the display doctrine in order to amplify the information content. Within this range the support of reaction asks for the most appropriate technique for interaction to ensure a quick and error-free response to the information stimulus without over-stressing the user.

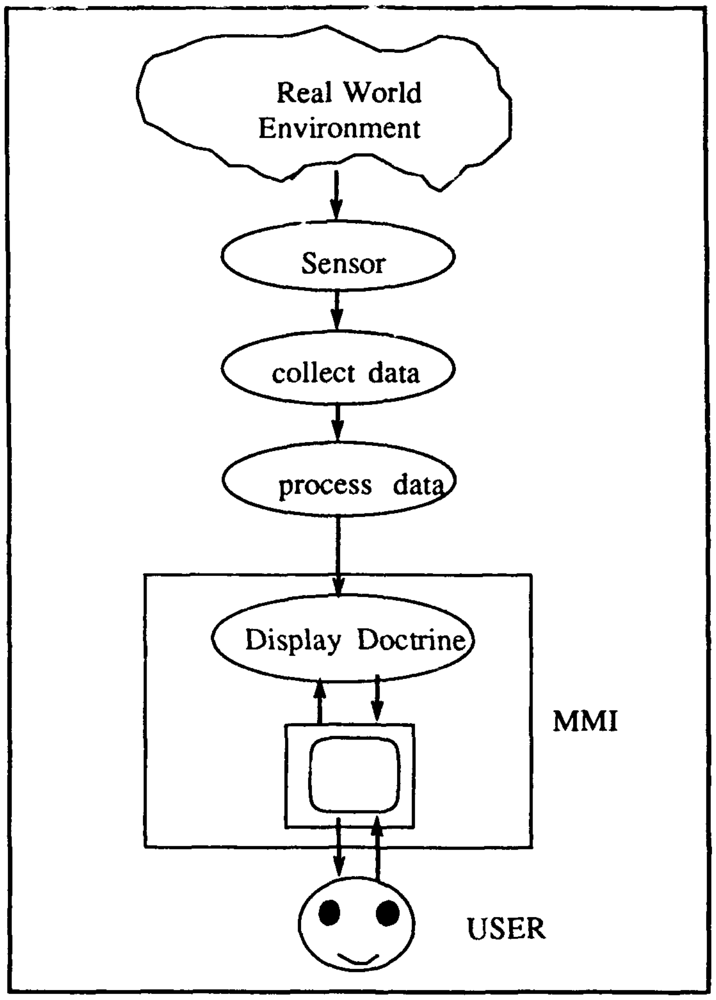


Figure 70: Simplified Model of Information Conversion

So, the other critical function for the MMI can be stated as:

(2) Provide a set of interaction techniques to optimize the possibilities for the user to feedback his decision into the system.

The design and development of a MMI with respect to recognition support will have to consider the concepts of visual and aural perception:

- which symbol is associated to what information,
- which sets of colors to use.
- how best to integrate aural cueing and alerting with the graphic screen representation.

Interaction techniques may be any one or combination of the following:

- commandline interpreters.
- menu systems.
- form filling systems.
- iconic systems.
- window systems.
- direct manipulation (activate a button by physical touch or position and selection using a pointing device).
- graphical interaction (drawing tools).
- speech.

C. COLOR USE AND SCHEMES

The use of color can be a powerful tool in the exchange and management of information on a visual display. For the tactical application color can be used in its qualitative sense, to show that one symbol, or class of symbols, is different from another. Color can impart special meaning to the symbol. This meaning can be arbitrarily assigned by the user or designer, or it can be selected to reinforce existing stereotypical information and assumptions. In any case, the key design issues for the MMI involve how much color to use, how much meaning to assign and how much user flexibility in color selection to allow.

The use of color requires care and restraint. Too much color blurs any meaning assigned and can drastically reduce user effectiveness. Effective color usage depends on matching physiological, perceptual and cognitive aspects of the human visual system [Ref. 21:p. 334]. Color is a sensation. It is perceived a little differently by everyone. People can identify a particular color within a range, yet disagree on shading and hue. These perceptions may change with age, prolonged viewing, brightness, context and size of the colored area. The cognitive issues involved with color, i.e. using color to express meaning, are greatly affected by the user model. When color is used to impart information based on user concepts already established through training standards, or stereotypical associations, (such as red for hostile tracks, danger, stop etc.) it can be quite effective.

Reference 21 provides some general guidelines for the effective use of color on visual displays:

- ***Do not overuse color.*** The benefits of color for imparting, organizing or signalling information are lost if too many colors are used. In consonance with the limits of human working memory, limit the display to about six clearly discriminable colors.

- ***Avoid the simultaneous display of spectrally extreme colors.*** Viewing extreme color pairs (such as red and blue or yellow and purple) requires constant refocusing and visual fatigue.

- ***Avoid pure blue for text, thin lines and small shapes.*** Blue does make a good background color and is perceived clearly out into the periphery of our visual field.

- ***Avoid adjacent colors which differ only in their amount of blue.*** Edges which differ only in amount of blue appear indistinct. This can make the display appear poorly focused.

- ***Avoid use of red and green in peripheral areas of large screen displays.*** Retinal peripheral sensitivity to these colors is poor. Blue and yellow are good peripheral colors.

- ***Not all colors are equally readable or legible.*** Extreme care should be taken in choosing foreground/background colors for text. Generally, the darker spectrally extreme colors (red, black, brown, blue) make better background while the brighter, spectrally centered colors (white, yellow) produce more legible text.

- ***Avoid the need for color discrimination in small areas.*** The human visual system produces sharper images with achromatic colors. For fine detail or small shapes and screen areas it is best to use black, white and gray. Reserve use of the chromatic colors for larger panels, backgrounds or for attracting attention.

- ***Group related functions or elements by using a common background color.*** A successive set of images (such as a series of overlapping menu windows related to a function) can be related by background color.

- ***Brightness and saturation draw attention.*** The center of the retina (fovea) only focuses on about 2 degrees of the visual field, the rest is peripheral. The ability to differentiate colors degrades as the object recedes into the peripheral visual field. The brightest and most highly saturated (purest) area of color will immediately draw viewer attention.

- ***"Warm" and "Cold" colors should indicate action levels.*** Traditionally, the warm colors are used to signify action or need for a response. Cool colors indicate status or background information. Most people perceive warm colors as advancing toward them, thus forcing their attention.

The above guidelines offer some suggestions to keep in mind when designing an MMI, but it should be noted that they cannot be considered binding for all applications. Once again, the development and use of an accurate user model which draws upon training, experience and stereotypical information pertinent to the user and environment in which the system will exist, cannot be overemphasized.

Each person perceives color slightly differently and while the guidelines for color use apply well in the general case, some support for individual idiosyncrasy is necessary. A suggested further constraint on the use of color would be to organize suitable color schemes into sets, allowing the user to choose from the available sets but not individual colors. This makes the users choice relatively simple and quick, and more importantly provides the system designers with the capability to enforce known human factors standards while still providing a range of preference for the user. This concept is used in the MMI for NCCS-A through use of controlled access (password) to a Workstation Configuration Utility which allows the user to change the default color scheme to at least one other optional scheme [Ref. 5:p. 29].

D. AURAL CUES AND ALERTS

Sound and the human capacity for hearing provide a powerful tool for human-computer interaction and communication. Aural cueing and alerting has already proven its usefulness in the tactical environment where the visual channel is close to overload.

> Human hearing, and the mental processes associated with it, have evolved to where verbal communication is perhaps <u>the</u> greatest means for information transfer. Pattern recognition capabilities of human hearing, however, are not limited to speech but include recognition of many special characteristics of both natural and man-made sounds. [Ref. 19:p. 422]

The concept of sound icons, aptly named "earcons", are particularly applicable to the LCCDS. Many of the sound features used to create earcons are easily implemented and manipulated on current workstations. Patterns of sound are easily recognized and remembered, even in the presence of high noise levels. The parallel processing involved with hearing, like that used in vision, provides nearly instantaneous recognition of sound patterns. [Ref. 19:p. 422]

The use of audio for implementation of system cues, alerts and warnings can greatly enhance the performance and reaction time of the user in an environment where parallel activities are also taking place. For instance, it is not unusual for the user in the Combat Direction Center (CDC) to be involved in internal/external communications, log-keeping and generating reports. The human eye can perceive the visual scene over a wide angle, almost to 180 degrees, however, visual detail is obtained only over a narrow region (about two degrees across), called the *fovea* [Ref. 20:p. 25]. The remainder of the retina provides peripheral vision for orientation. This limitation can drastically dilute the impact of visual cues in an environment where the user is not *always* focused on the display. Sound icons may transmit meaningful "chunks" of information much faster than textual messages displayed visually or spoken via computer synthesized voice. There are four major advantages of audio cueing [Ref. 19:p.422]:

1. Users need not pay strict attention to audio cues, relying on subconscious processes to detect significant events, while focusing on more interesting tasks.

2. The user need not be within clear line-of-sight of the display. Distance and orientation to the source of the sound is completely arbitrary.

3. Well designed audio cues may be easier to learn than certain other forms of communication such as visual icons.

4. The visual landscape on a graphical display is already cluttered, and audio cues would limit the number of distracting textual messages.

The LCCDS shall provide the user with a MMI heavily dependent on visual display of complex data and spatial relationships. Use of audio cues to augment the intuitive and fast exchange of information and control between man and machine has a number of constraints as well as advantages. Deatherage [Ref. 22] provides some basic guidelines as follow:

Use auditory presentation if:

1. The message is simple.
2. The message is short.
3. The message will not require the user to remember it for later reference.
4. The message deals with events in time.
5. The message calls for immediate action.
6. The visual system of the user is overburdoned.

E. SPEECH RECOGNITION

Speech recognition systems do exactly what their name suggests; they recognize the user's spoken words. The words are typically recognized using pattern matching techniques against a set of "acoustic templates". Once the word is recognized it is passed to some higher level software routine for syntactic and semantic analysis.

It is important to make a clear distinction between speech recognition and speech understanding. A speech recognition system has no understanding of the words it "hears". It simply matches the acoustic pattern of the sound against a set of predefined templates. Once a match is made, higher level software attaches meaning as specified by the system developers.

There are four generic types of speech recognition systems currently available and being used widely by industry. These can be split into two groups; (1) speaker dependent or independent, and (2), connected or discrete speech (either may be speaker dependent or independent). The following definitions are taken from Poock [Ref. 23].

1. Speaker Dependent

A speaker dependent system will reliably respond to the voice inputs of only those users who have previously "trained" the system. In this context, "training" the system refers to the process of creating a set of acoustic templates for each user. Dependent systems come equipped with special software which allows the user to define his own vocabulary, then provide speech samples to the system. These samples are used to form the templates. In other words, a dependent system needs to hear not only the right words, but the right voice as well. When the system (generally, for most systems) has what it considers to be a good sample, it will notify the user that it has been adequately trained. At this point the system is ready for actual use. These systems require that each user "log in" when ready to use the speech recognition system so the proper set of acoustic templates can be loaded into memory.

Proper training of the speaker dependent system is of the utmost importance in assuring reliable recognition of the vocabulary for any particular user. Poock [Ref. 23] at the Naval Postgraduate School is conducting research in the areas of speech applications and how to most effectively train speaker dependent systems for optimum reliability and performance.

2. Speaker Independent

A speaker independent system contains more complicated algorithms which enable it to recognize many different voices and dialects. These systems require no previous "training" by prospective users. They should be able to recognize anyone who tries to use the system.

Theoretically, the speaker independent systems are not as reliable as dependent systems, but both systems are showing recognition accuracies in the 97-99% range for vocabularies of several hundred words (for the normal office environment). [Ref. 23]

The next two areas of speech recognition are concerned with the way in which the user speaks to the system. Both Discreet and Connected may be used with either speaker dependent or independent systems.

3. Discrete Speech

A discrete system contains a given number of sound patterns which the user may speak. A sound pattern may be a single word, or phrase which is associated with a single acoustic template. When using the discrete system, the speaker must pause between each word or phrase (utterance) he makes. Depending on the system, this required pause is normally in the range of 100-250 milliseconds. When the system "hears" the pause, the utterance is ended and the system searches the available templates for a match in order to determine what was just said.

4. Connected Speech

Connected speech requires no pause between utterances. These systems contain an algorithm which determines word boundaries. Connected speech, especially for digit-type strings of input, is much more natural for the human user than discrete speech input [Ref. 23]. Some existing connected speech systems are especially impressive in that they continuously monitor what the user says recognizing valid connected speech portions and ignoring the rest.

5. Applications for Speech Recognition

There are very few things a human can do concurrently. Physical and cognitive limitations preclude doing things in parallel. Under certain circumstances however, speech can be an exception. Speech, in conjunction with other visual or graphic oriented activities is a common occurrence. A common instance is holding a conversation while driving your car. Another could be positioning the Ball Tab over a track symbol on the TacPlot while invoking menu selections made and executed via speech input. Speech input is useful when: [Ref. 23]

1. Eyes are busy.
2. Hands are busy.
3. Darkness or low light levels exist.
4. You wish to enter data but your hands are not in a position to touch a keyboard. You could use either a hardwired or wireless mike for data entry, even though the distance between user and keyboard is small.

6. Conclusions on Speech Recognition

It becomes obvious, once the capabilities of speech recognition are known, that there are many applications for the tactical environment. More research in the further development and tactical application of speech I/O is necessary. Poock [Ref. 23] points out that the biggest stumbling block with technical applications of speech I/O is that people are just not aware of what it can really do for them.

F. AN EXTENSIBLE MMI

Any interface that permits additional functionality or capability to be included in an existing system, as the system is used and new applications and ideas develop, can be considered extensible. This concept covers a broad range of capabilities whose primary purpose is to handle varying levels of user experience and build in some support for future growth. These capabilities cover a broad range, from keyboard macros, or "hot keys" in lieu of menu selections, to the ability for the user to define new functions. The ability to incorporate new methods and ideas into existing form and function, quickly and easily by the user himself, can have a major impact on the efficiency and reliability of tactical decision tools such as the LCCDS. Two suggested starting places for extensibility are in the "display doctrine" and the Tactical Database.

The display doctrine (defined in detail in Chapter II) is that set of rules and tactical parameters the system applies to the processed data to transform or filter it into usable, comprehensible information. Much of the display doctrine will take the form of existing NTDS data display standards. However, it should also provide the user with the capability to create his own functions for data manipulation, more focused on his specific requirements. The ability to do this could provide endless applications for specific mission requirements, tactical doctrine, navigation safety, etc.

The Tactical Database shall hold all the data related to NTDS standard symbology, associated data, and Special Points created with the workstation enhanced graphics capability. In support of MMI extensibility, this database should allow the user to identify newly created graphic objects, symbols and associated data as valid objects to be recognized and stored (if desired) as part of the Tactical Database. Additionally, once a user created object has been incorporated into the Tactical Database, the user should have the capability of assigning or defining the operations allowed on it via the Display Doctrine.

An additional, more complex form of extensibility is the concept of *adaptive* interfaces. Simply put, it means the MMI should adapt to the user rather than the user adapt to the system. There are two forms of adaptive interface. One way is to allow the user to make his own modifications to the MMI if he finds the behavior of the system unsatisfactory once he has used it. The idea is that the MMI may be changed by several classes of user. These classes cover the range from novice to expert. The amount of MMI change allowed is dependent on the user who wants to make the change and the access privileges he has to the internals of the interface. This can produce a better interface, but it does leave the burden of adapting to the user [Ref. 24:p. 2].

The second, much more complex, form is *dynamic* adaptation by the system itself. In this case the interface changes with respect to particular user and current context. The MMI learns with respect to each individual user. This implies the interface may work differently for each user within the same task context. For optimal system performance, the dynamically adaptive MMI should be able to compensate for the inherent biases of the user [Ref. 24:p. 3].

Along with the obvious power and friendliness, there are a number of negative aspects associated with dynamic interfaces. They are complex and require a lot of system resources, but more importantly, the user may not be able to develop a coherent model of the system if it changes frequently. This may undermine the users confidence and performance with the system. If the user does not have a clear understanding of the system behavior his effectiveness can be seriously reduced. [Ref. 25]

We have attempted to incorporate the first method of adaptation, letting the user make his own modifications, in the LCCDS. The desired level of user friendliness flexibility can be supported while at the same time minimizing any risk of violating the users conceptual model of the system. The model only changes when the user decides to change it. This concept of adaptability falls within the NAVSEA guidance [Ref. 4] for the LCCDS and the SPAWAR NCCS-A Workstation Executive specification [Ref. 5].

V. CONCLUSIONS

A. RECOMMENDATIONS

1. Formal Approach

This thesis is intended to provide the first stage or activity within the software development cycle as outlined by Berzins [Ref. 6]. Chapter II provides the goals and constraints of the LCCDSWS representing the initial requirements.

However, not all goals could be refined as necessary. In particular goals 4.4 and 4.5 could not be refined sufficiently since the interface descriptions are not yet available and will have to be added at a later time.

In the same way we do not expect the goal hierarchy to be complete. Further refinement of goals, addition of new subgoals or reorganization of the hierarchy might be necessary as knowledge on details is gained in further stages of the development cycle.

The goal hierarchy is not yet completely defined, but the requirements analysis can be continued now by providing *functional specifications* for the proposed system. The goal of this stage is to precisely define the external interfaces of the system. The combination of initial requirements and functional specification is referred to as the final requirements [Ref. 6: p. 3-1]. Chapter III provides a high level definition of system processes and data objects necessary to support communication between LCCDSWS and the external interfaces. The next activity in the software development cycle is the architectural design which decomposes the system into software modules. And finally, the goal of the implementation is to construct a program that correctly realizes the specified interfaces for each module identified during architectural design and meets the associated performance constraints [Ref. 6: p. 1-9].

The concept of, and relations between, the different stages of the software development cycle are shown in Figure 2 (page 8). Down-arrows show the normal flow of execution, up-arrows represent details gained at later stages, which require the repetition of an earlier step. The large loop labeled "Evolution" demonstrates that the software system is subject to changes due to altered operating conditions or user needs, resulting in a new version of the system. The main difference between the initial development and later evolution is that many parts of the existing version can be re-used in the next version [Ref. 6:p. 1-4]. We propose that these steps, applied to the LCCDSWS development, represent the baseline to be carried out via further thesis work at NPS.

2. Rapid Prototyping

During the development of software systems, especially hard real time systems, a worthwhile effort to increase software development productivity is rapid software prototyping. The idea behind rapid prototyping is to create a prototype of the proposed system to verify that the real time behavior demanded by a customer is feasible under the imposed constraints. This can save tremendous amounts of resources in terms of time, effort and money. The feasibility of a system is verified before commitment to an actual implementation is made. Design errors can be detected more easily and are cheaper to correct at this point (*especially* compared to redesigning and recoding a finished product which does not meet the requirements).

One such system for rapid prototyping is CAPS (Computer Aided Prototyping System), which is based on PSDL (Prototype System Description Language). CAPS is presently under development in a research project at NPS [Ref. 26, 27, 28] and can be characterized as a composition of separate tools which provide the means to create a prototype of a software system. CAPS will provide an integrated environment for the development and testing of prototypes of software systems. It is especially of interest for LCCDS because the system will automatically translate the PSDL descriptions of a system into Ada code and compile them so that they could be executed to demonstrate the prototype. CAPS is specifically designed to address large embedded systems which have hard real-time constraints and employs a database system to store and recall reusable software components in the Ada language.

The requirements definition process is the most difficult and error prone phase of software system development. The use of informal requirements definitions, rapid prototyping systems and formal design languages, such as PSDL, *together* work to bridge the conceptual gap between the users desires, designers concepts and final implementation. The importance of the prototype is that it provides the capability for generating "quick-and-dirty" samples of eventual system functions and capability which the user/customer can see and feel. A rapid prototyping system can provide the vehicle, *early* in the requirements definition phase, for uncovering design errors, inconsistencies and misconceptions which might otherwise grow into costly design or maintenance changes later in the software life-cycle.

We recommend to use CAPS to study the real-time behavior and feasibility of the LCCDSWS as soon as CAPS becomes operational.

3. Further Research

One of the characteristics of Berzins' formal software engineering approach is the concurrent nature (hence the bi-directional arrows in Figure 2) of these activities. This provides an opportunity to work on different aspects of the project simultaneously. One of these aspects is the use of commercially available software packages as suggested by NGCR [Ref. 3]. These packages are tested and debugged through "best commercial practice". With developments amortized by thousands of commercial users considerable reduction in both development time and cost can be achieved. In the following paragraphs we provide some examples for such packages as they may apply to the LCCDS development.

a. Commercial Databases

Increment One of the LCCDS requires the selection of a suitable object-oriented database management system for the project. Ross [Ref. 16] evaluates three possible systems and shows that GemStone, a product of Servio Logic Corporation, Alameda, Ca., best fits the requirements of this project. To determine the usefulness of this package it is necessary to install this database on a UNIX OS and implement an interface to the Ada language as suitable for the LCCDSWS.

b. Real-Time Operating System

One of the drawbacks in using the UNIX operating system for the LCCDSWS development is that it does not provide real-time mechanisms. To make the step from the prototype to an operational system it is necessary to select one of the commercial available UNIX-derived operating systems, which provide these mechanisms and install it on the proposed Sun Microsystems Sparcstation 1. It is expected that software for the project developed on UNIX will retain compatibility.

e. MMI

The Man-Machine-Interface will be one of the most complex modules within LCCDSWS. Implementation of this interface using commercial available software tools should be considered. The X Window System is a network-oriented, server-based windowing system developed at MIT. It is supported by most vendors of UNIX-based workstations and is becoming a standard in this environment. Applications will usually use the Xlib toolkit and widgets, software tools that include a text editor, menus, scroll bars, icons and command buttons. X Windows is available at NPS, some research has already been carried out with respect to the LCCDSWS. X Windows can be used to implement the MMI for LCCDSWS once an interface to the Ada language has been written.

An important option to explore is the use of existing software for the NCCS-A Workstation MMI currently under development at NOSC. The OPNAV sponsor for NCCS-A, OP-942F4, has offered NPS full use of all NCCS-A MMI object code for incorporation into LCCDS [Ref. 29].

d. Reusable Software

Another finding of the research for LCCDS is the existence of general purpose Ada software packages on the SIMTEL20 software repository. This repository contains a variety of generic data structures and algorithms, software engineering tools and highly specialized applications. An example for reusable software is the Kalman-Tracking-Filter package contained in SIMTEL20 which provides real-time tracking algorithms. The SIMTEL20 software is available at NPS (and for anyone else on DDN) and should be explored in depth to identify packages that can be used for the LCCDSWS development in particular and the software development life-cycle in general.

e. *Rewrite existing Software in Ada*

The NGCR report recommends serious consideration of rewriting existing software into Ada [Ref. 3 :p. 2]. This in particular is a promising approach for LCCDSWS since all of the CDS functions as required in goal 4 are implemented in the CMS-2 language for currently operational CDS's. These systems should be analyzed to determine which algorithms would be both useful for the LCCDSWS and feasible for an Ada rewrite.

B. EVOLUTION OF THE LCCDS

Evolution is the process of modifying or extending the functionality of a software system. It may be a response to changes in user requirements or part of a planned phased development of a system as a series of increasingly sophisticated versions [Ref. 6:p. 1-10]. The initial requirements stated in Chapter II represent a detailed refinement of the "user" high-level requirements outlined in the NAVSEA Statement of Work [Appendix A]. In this chapter we envision the evolution of LCCDS in terms of two sets of possible enhancements to the system. Section A describes modifications to the basic version of the LCCDSWS. Section B concerns functional extensions of LCCDSWS.

1. Modifications to the Basic LCCDSWS

The functionality described in the Chapter II requirements form a small subset of current CDS capability. The primary purpose of these requirements is to define a small scale, *prototype* LCCDSWS. Once the system has been developed to the point where some performance and functional evaluation can be done, perhaps after

Increment One is complete, it may be appropriate to consider some greater functional capabilities. We propose several enhancements to the LCCDS in the following paragraphs.

a. Trial Maneuver

This function was brought to our attention by Surface warfare officers who have used the Raytheon Collision Avoidance System (RAYCAS) installed on the bridge of some Naval combatants. The system we looked at was installed on the USS Ranger in San Diego, CA. RAYCAS provides an easy-to-use digital MMI directly superimposed over a Radar repeater. NTDS-like symbology can be imposed over digitized Radar video. RAYCAS operates in one of two modes; True or Relative. True mode displays all contacts with speed leaders oriented to true course and speed through the water. Relative mode displays all contacts with speed leaders relative (to ownship course/speed) course and speed. In addition to providing the usual information regarding course, speed and CPA, the RAYCAS also provides a function called "Trial Maneuver". This function greatly enhances the user's ability to visualize the relative motion between moving objects and Ownship. Further, it allows for insertion of trial course and speed alterations showing the user their overall effect on all other displayed surface contacts *before they actually maneuver the ship*. This has proved to be immensely useful, especially when navigating congested waters.

b. Optimum Route Planning

Optimum route planning refers to the capability of specifying some (possibly large) number of conditions which affect the route by which a vessel may proceed from one point to another. In a tactical environment the constraints on movement from place to place become extremely complex. A large number of factors must be taken into account for transit of naval vessels. For example, necessity for avoiding detection, position and plans of other battle group units, rapidly changing intelligence information, satellite and aircraft reconnaissance coverage, etc. all must be considered.

Equally complicated is route planning for CLF vessels conducting "delivery boy" operations for fleet support underway. A CLF ship is normally tasked to rendezvous with several ships in a battle group for purposes of underway replenishment (UNREP). The task of planning an efficient UNREP schedule is difficult and complicated, involving analysis of current and projected positions of several ships, distances between and time alongside each ship.

This situation is similar to the "travelling salesman" graph problem with the exception that all the "cities" are moving. The application of graph analysis and artificial intelligence (expert or "rule based" systems) to help determine optimum routes for complex situations could be extremely beneficial for efficient tactical and logistic planning.

c. Speech Input

The use of speech recognition software could be used to solve some of the harder problems associated with the LCCDS MMI and other interfaces. There are difficulties associated with supporting a wide range of user expertise, interfacing LCCDS with existing shipboard systems (primarily radar), and even deciding between use of a trackball over a mouse.

Speech recognition as an input medium may apply to a large number of LCCDS functional areas. Several are discussed in the following paragraphs.

d. System Control

LCCDS uses menu-driven system control. The user will spend most of his time in dialog with the system via mouse/trackball/cursor controlled menu selection processes. It would be much faster to use speech recognition and let the user simply tell the system what he wants. The menus, also activated via speech input, would still be available to aid the user by providing available options.

e. Trackball Functions

The use of a mouse or trackball driven cursor to identify objects (tracks, menu choices, windows, etc.) for program action requires a button, or set of buttons, which you "click" or "drag" as necessary. This works okay for a mouse, but is hard when using a trackball. To use a trackball the user really must have both hands free, but this can become very clumsy and inconvenient when he has something better to do with one of his hands (such as manipulate keys on his keyboard). The mouse doesn't present this problem, but does have other, possibly more serious problems. You must have a fairly large space to run it around, and from a maintenance perspective, it may not be suitable for the harsh shipboard environment.

Speech recognition could be used, in lieu of physical buttons, with the trackball to provide the click/drag function while the user positions the trackball. The cursor is positioned using one hand, objects pointed to are identified verbally, and the user's other hand is free.

f. Radar Interface

The Radar interface must extract the required positional information from the raw video provided by the Radar system, digitize or format it, and send it to the LCCDS. Hardware implementation of this will be costly and time consuming. It is possibly to provide raw radar information directly to the LCCDS using speech recognition.

It is Navy SOP to have an operator always available to analyze and report the radar picture. This operator is on a sound-powered phone circuit with bridge and CDC personnel, providing radar contact information in a standard format for all to use.

With the addition of a microphone, appropriate for use in a speech system, hardwired to the LCCDS, this radar information could be input directly into the tactical database just as if the information had come across an actual hardware radar interface. The radar operator passes information as he normally would, only now it goes directly into the system as well as up on the various status boards.

2. Extensions of the LCCDS Functionality

a. Training Function

People who interact with a software system need training on proper use of that system. Users need to develop, maintain and expand on their conceptual model of the system. They need to enter data, fill in forms, make queries, interpret system responses and so on. Training is necessary to increase the *Operational Readiness* of a CDC team. Training for a ship-based combat direction system can be carried out while a ship is at sea, with all subsystems like navigation, radar, Link 11 etc., fully operational. The "real world environment" provides the scenario an operator can be trained on. However, it is most likely that operator training will be necessary in a situation where some or all of the subsystems are not available, either at a shore-based training facility or actually onboard the ship (i.e., during maintenance periods, operational standdowns, etc.). The Training Function will provide the capability to train operators in such a situation by software simulation of non- operational subsystems and a possibility to establish an artificial scenario for training purposes.

Scenarios can be divided in the following exercise categories:

- Navigational Practice: the operator is confronted with a certain navigational situation where he has to ensure safe passage using the system.

- Anti-Collision Exercises: the operator has to use the system to carry out anti-collision procedures to avoid collisions with simulated surface tracks.

- Safe ship-handling and formation maneuvering exercises.

- Link 11 Practice: the operator has to carry out Link 11 track management.

- Combat Exercises: the operator is confronted with simulated air, surface or subsurface attacks or a combination of those, including simulated missile and gun exercises.

- Combinations of the above for more sophisticated scenarios.

Scenarios can be either standard predefined "canned" evolutions for each training session or coordinated exercises involving more than one workstation and student. Eventually, these LCCDS's should have all the necessary training software support "built in" from the initial design. The benefit of this is more onboard operator training capability and less dependence on scarce shore-based facilities.

b. CDS Functions

(1) Weapon Interfaces. One of the most important features of a combat direction system not currently supported by the LCCDS is the interface to a weapon system. The need for such an interface is certainly dependent on the type of ship LCCDS is employed on. Non-combatant CLF ships, possible target platforms for the

LCCDS, are not equipped with weapon systems. However, the availability of LCCDS on a non-combatant might imply the need for such a system as the next logical step. Possible candidates are systems for Anti-Ship-Missile-Defense such as CIWS or RAM (Rolling Airframe Missile). For small combatants (PHM's and amphibious craft) an interface to the Harpoon Weapon System or even a fire control system is possible. The interfaces should provide the operator with the capability to employ the weapon systems in their various operational modes.

(2) Automatic Multiple CPA Function. The CPA function of the basic LCCDSWS calculates the CPA of a track on user request. It issues a warning of a Close CPA or Collision CPA as appropriate when the track is within certain specified time and range criteria. The Automatic Multiple CPA Function should calculate CPA values for all surface tracks within a specified range and issue warnings when the criteria are met automatically. It should be possible to enable or disable this function.

c. Simulation Functions

Simulation Functions should provide the capability to simulate external subsystems of the LCCDS. They are useful in two ways: first they allow extensive system testing in a situation where the use of external subsystems is not possible or not appropriate. Second they support the training functions as described above. The most important characteristic of simulation functions for LCCDS is the need for hard real-time behavior of these functions. We propose the following set of simulation functions for LCCDS:

- Simulation of a navigational system;
- Simulation of a radar system;

- Simulation of a passive Link 11 system;

- Simulation of weapon systems as appropriate.

3. LCCDS Future Versions

Based on the above modifications and extensions we can now provide some considerations on possible future versions of LCCDS employed in various roles.

a. LCCDS on Non-Combatants

The user requirement for LCCDS as stated in the NAVSEA Statement of Work [Appendix A] demanded a system that can potentially be installed on non-combatant ships. A possible configuration is shown in Figure 71.

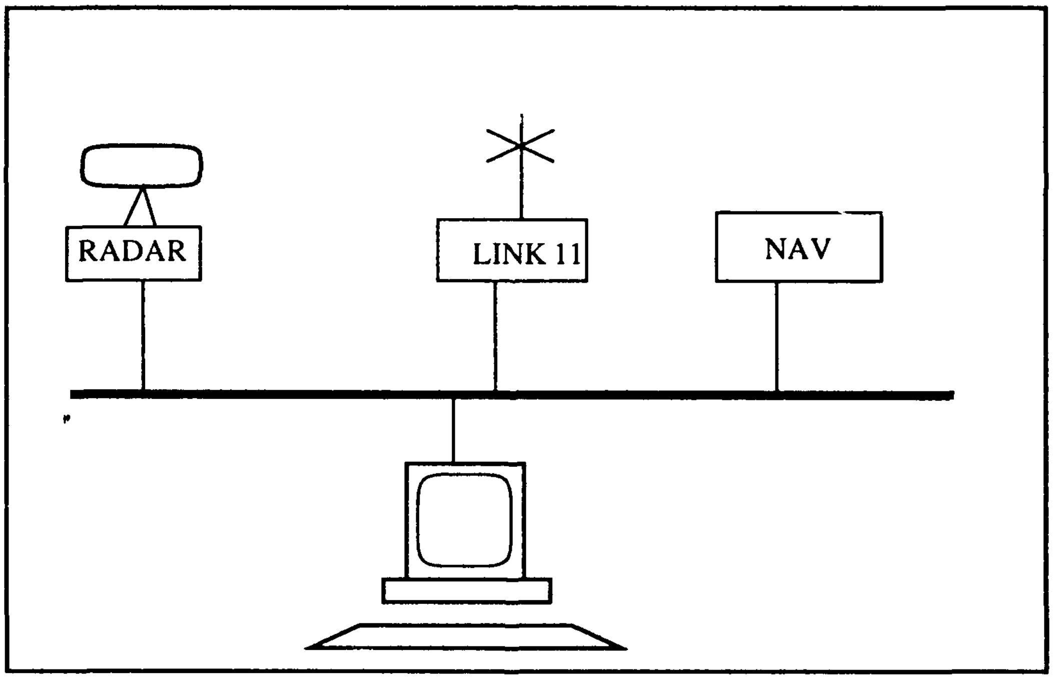


Figure 71: Non-Combatant LCCDS

It can be expected that the availability of a low-cost "mini" combat direction system for CLF ships may imply the desire for an air self-defense capability for these units by installing a close-in air defense weapon system (CIWS)such as Phalanx. Figure 72 depicts such a configuration.

b. LCCDS on Small Combatants

Combat Direction systems have been the domain of large combatant units primarily because bulky (and *very* costly) equipment made an installation on small

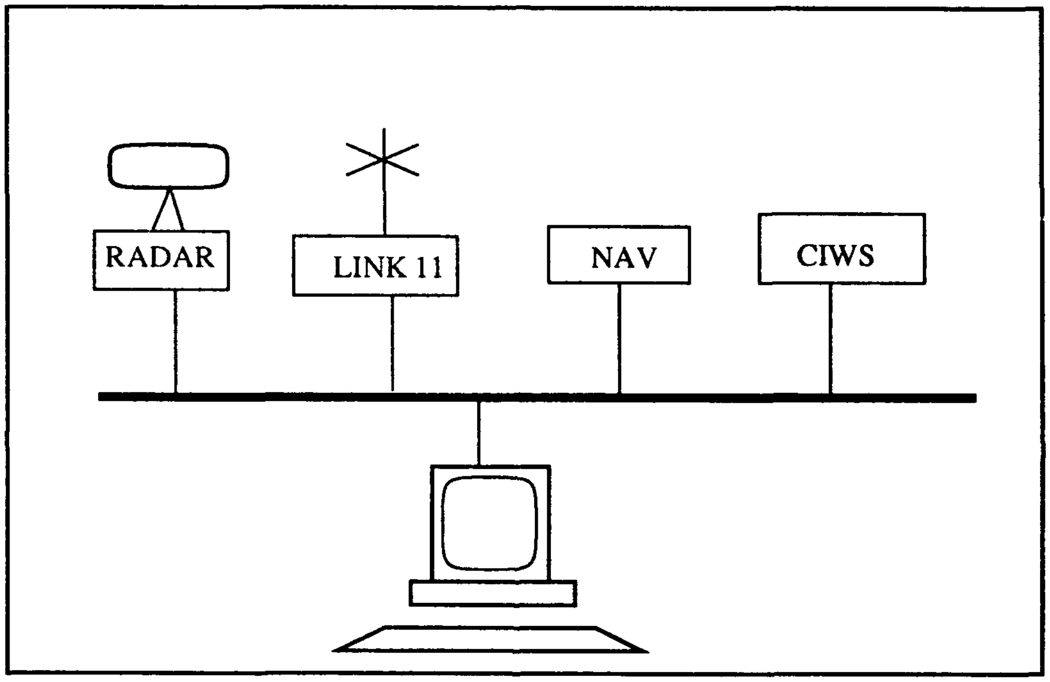


Figure 72: Non-Combatant LCCDS with CIWS

combatants impossible. For PHM's, minesweepers etc., LCCDS can be a feasible and effective upgrade. LCCDS, adapted with the necessary interfaces to weapon or fire control systems as appropriate may look something like Figure 73.

c. LCCDS on CDS Equipped Ships

In addition to the installation on non-combatants, the NAVSEA Statement of Work states the desire for installation on current CDS equipped ships to augment the existing processing capability. We propose the utilization of a LCCDSWS as a tactical workstation serving in a particular role. For example workstations could be used to support individual elements (e.g.,ASMD, EWC, etc.) of the warfare task.

The tactical workstations can form a separate network which can be interfaced to NTDS via gateways. Tactical workstations employed in highly specialized roles are expected to significantly reduce the workload of the central

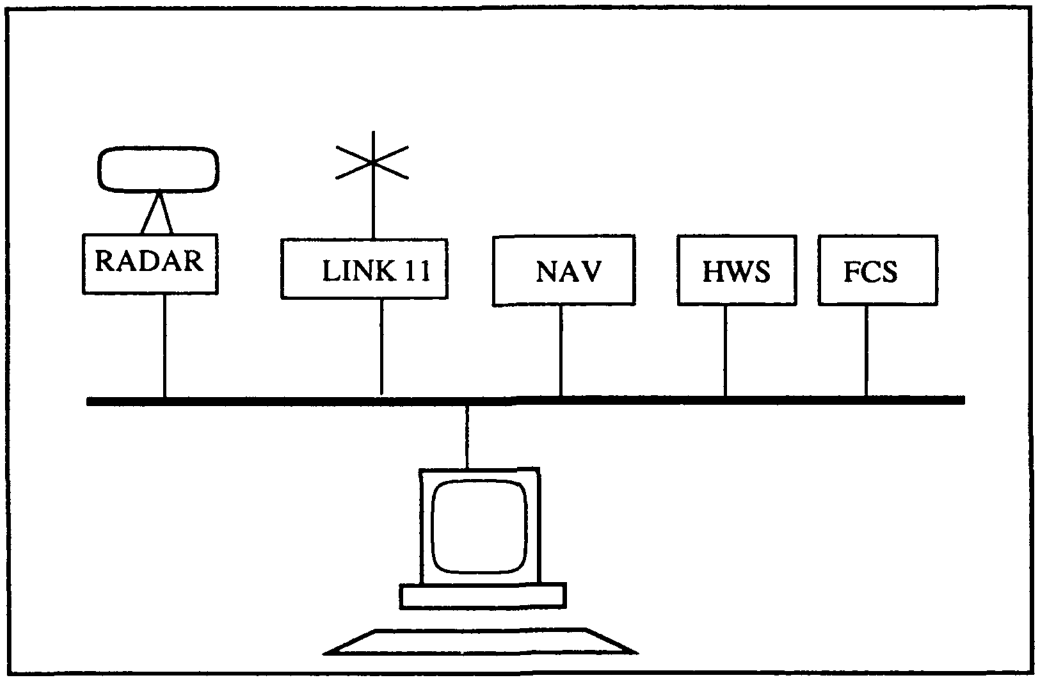


Figure 73: Small-Combatant LCCDS

processor (Figure 74). The graphics capability of the workstations will not only improve the way in which tactical significant information is represented, it also opens a whole new area of applications for tactical information management and access. For example, tactical documentation like the TAO manual could be made available through alphanumerical windows. Graphical evaluation and processing of satellite photos for weather or intelligence purposes is also possible. In-depth human factors research on the methods for management, organization and display of complex tactical information in the workstation environment is being conducted by Osga [Ref. 30] at the Naval Ocean Systems Center (NOSC) in San Diego.

d. A CDS Based on Distributed Architecture and Tactical Workstations

One of the dangerous bottlenecks of existing CDS is the centrally located CPU of these systems. The breakdown of this unit reduces the capabilities of the CDS significantly and can result in a total loss of the current tactical picture. The

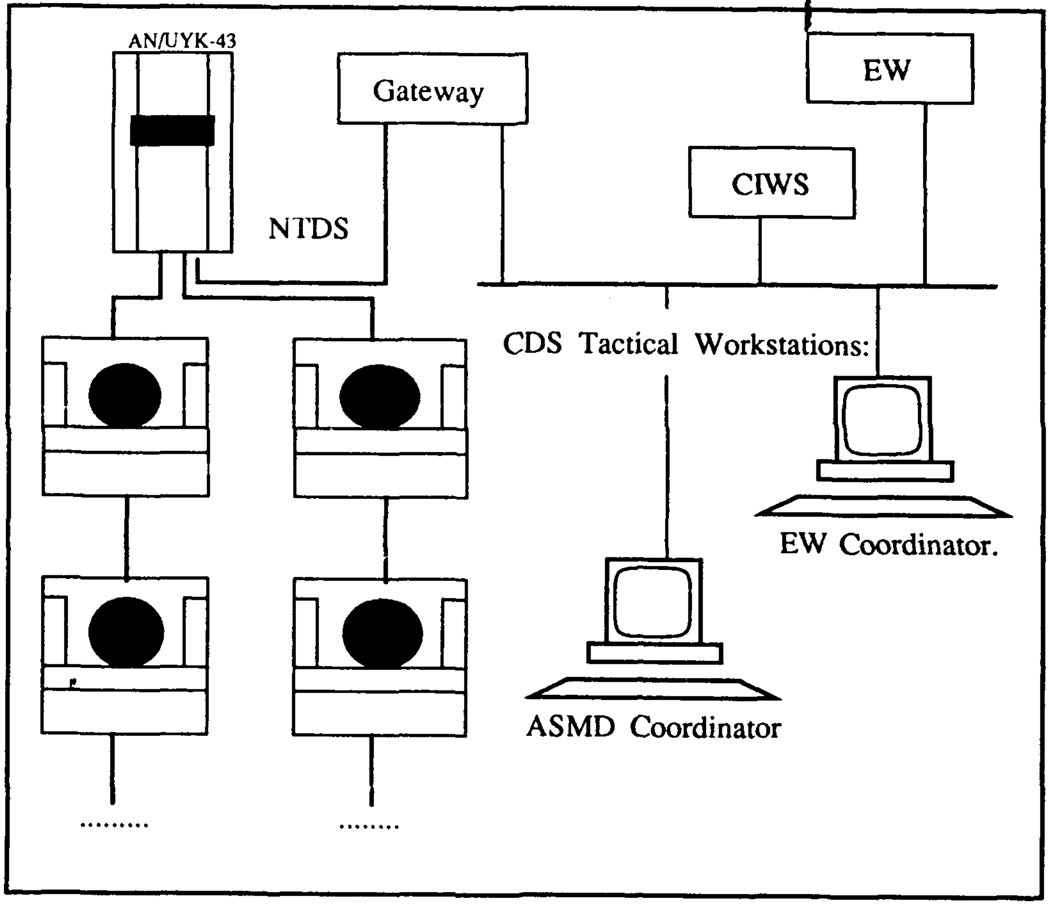


Figure 74: LCCDS as CDS Tactical Workstation

replacement of the CPU by networked workstations (Figure 75) provides a variety of benefits. The first is improved system survivability.

A failing workstation can not affect the functionality of the network or any other workstation. It's tasks can easily be taken over by another station or a spare station until the failing unit is replaced. Thus the breakdown of one or several components reduces the vulnerability of the system to hardware failures in a lesser degree.

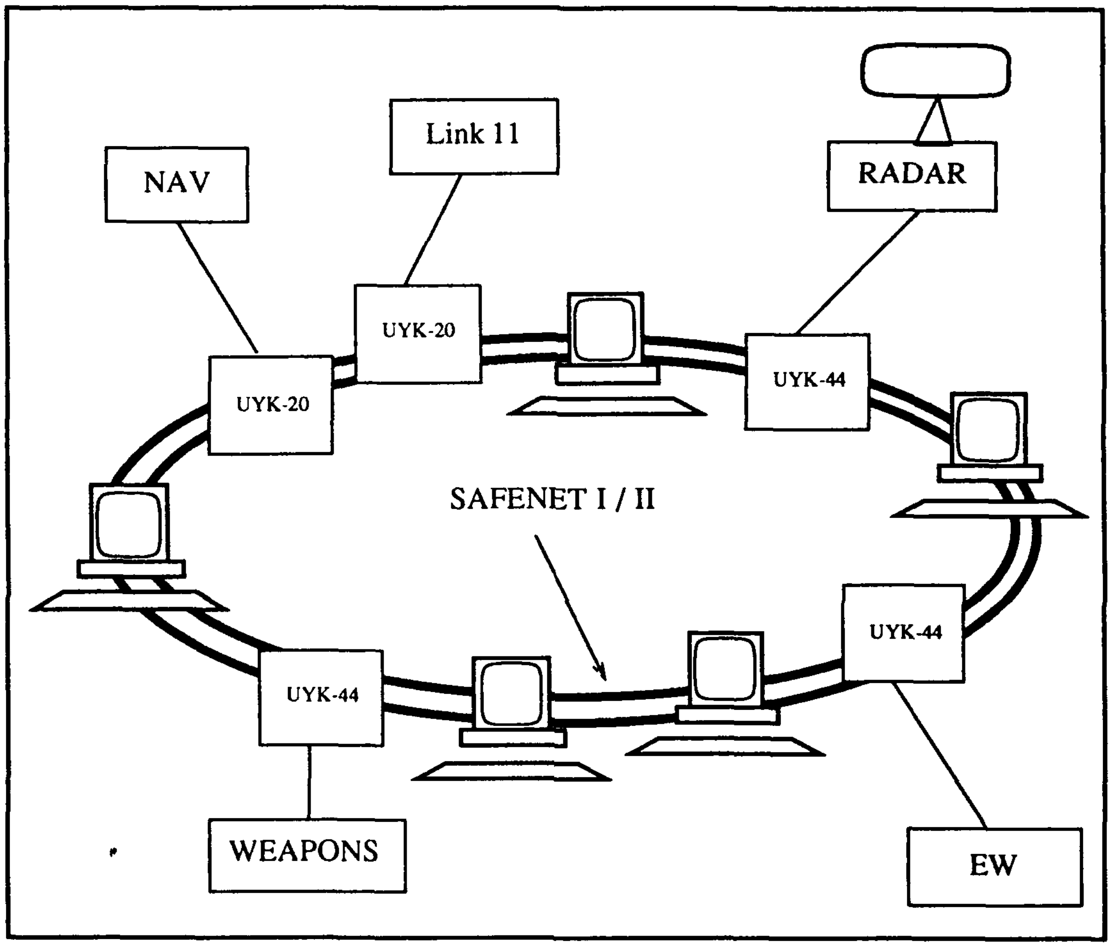


Figure 75: Future CDS - Distributed Architecture Using Tactical Workstations

Other aspects are cost and performance. LCCDS can here provide what its name implies: a low cost system but with increased system performance! A rugged SOLARIS SPARCstation 300E provides 16 MIPS and up to 56 MBytes of main memory at $90,000 compared to 3.5 MIPS, up to 5.2 MBytes and $2,000,000 of a AN/UYK-43 [Ref. 31].

The implication is that use of a distributed, "open system" type of CDS architecture can be beneficial is a wide range of areas. These include both purely tactical issues such as enhanced user interfaces, survivability, modularity of resources, reliability, ease of maintenance and training, as well as a host of new applications yet to be defined but now feasible due to the superior technology of the bit-mapped workstation display.

C. LCCDS FINAL CONSIDERATIONS

One of the most serious problems that has faced the weapon system acquisition process is the lengthy period of time (10 years is not uncommon for large systems) between initial requirements analysis and delivery of a useable system. As a result the computers associated with these systems are usually obsolete years before the rest of the system is finished.

While there may be no reasonable way to speed up the acquisition process for large systems, we can avoid being stuck with expensive, antiquated computers by delaying the acquisition of computing resources until the very last stages of system development whenever possible. The system design should be modified, or left open, throughout development to take advantage of rapidly improving commercial computer technology, hardware and software. Specifically, we should take advantage of the more advanced, reliable and less expensive commercial workstation systems. The savings seen in hardware costs alone could open new avenues of software development for efficient use of current computer technology.

The main point to be made here is that these workstations are not just sitting on someone's "drawing board". They exist and are available in ruggedized versions for immediate use for military applications. The holdup will be providing reliable software which not only supports the requisite CDS functionality, but also makes efficient use of the immense power provided by the hardware.

The methodology for software development is steadily improving. Better software is being written largely due to increasing usage of formal modeling of system architecture, computer aided software engineering (CASE) tools, rapid prototyping and reusable software. In our opinion the existence, and rapidly growing availability, of these tools and methodologies will significantly streamline, economize and speed up the software development process. The key is to develop *reusable* software using formal design methods and automated tools.

There are several advantages to this reusable software approach in addition to the basic development time/money savings inherent in reuse of existing systems. Different systems can provide the operator with the same "look and feel" via a shared MMI. Experience with one system becomes useful for cross-training on many other systems. Configuration control can be enforced by making the software available as a toolkit, i.e., object code is available for use in any desired application, but the toolkit itself cannot be tampered with.

The LCCDS would be a great project on which to apply the above principles. There is a lot of related or parallel effort in the area of command and control software. In the case where the different systems use compatible hardware, they should also use compatible software for implementation of the similar functions. LCCDS and NCCS-A serve as a good example.

The LCCDS and NCCS-A systems manage the same basic data objects and share many similar functions. In our view the central LCCDS issue is the MMI, at least for the early phases of development. While there are some critical differences between C^2 and CDS systems such as real time performance and control of external systems, there seems to be no reason why the MMI for each could not be identical, and obvious economic and training related benefits if they are identical.

Specifically, NCCS-A and LCCDS will be developed for microprocessor based workstations with extensive use of commercial software. Much of the software already developed for NCCS-A is directly applicable to LCCDS requirements, most notably the MMI.

APPENDIX A

In order to provide a more complete picture of the Low Cost Combat Direction System project we include here a copy of the original Statement of Work issued as Enclosure 1 to the NAVSEA letter [Ref. 4] which initially introduced NPS to the LCCDS concept. We used this document as the cornerstone of our requirements analysis for the LCCDS.

Enclosure 1

STATEMENT OF WORK
for
LOW COST COMBAT DIRECTION SYSTEM (LCCDS)

Background

Combat Direction Systems have been in an evolutionary development cycle since 1958. Early systems provided basic ship to ship data link capability, analog to digital conversion of tactical data, and manual, rate aided, tracking. Todays systems have progressed to a level of sophistication that includes upwards of 20 tactical interfaces including multiple tracking sensors, multiple weapons interfaces, electronic warfare and multiple tactical data link systems. Along with this increased capability has come increased processing requirements and the need for greater sophistication in tactical display capability which in turn has greatly increased the cost of fielding new Combat Direction Systems.

Recently the Navy has initiated what is called the Next Generation Computer Resource (NGCR) program which is a cooperative effort between Navy and industry to field a set of state of the art computers for shipboard use in the late 1990's. Compatibility between the various computers will be through standard protocol rather than through common instruction set architecture or form-fit-function replaceability. Although the standards for NGCR computers are not yet fully specified, features will include the following:

a. 32 Bit ISA

b. 1 - 100 MIPS performance depending on application requirement

c. 26,000 hour Mean Time Between Failure

d. One or more of the industry and government back plane bus standards including VME, IEEE 896 (FUTUREBUS), IEEE 1296 (Multibus II), and VHSIC PI-Bus

e. Three types of local area networks to choose from depending on the application requirement. These include: SAFENET I (<10 Mbps), SAFENET II (>100 Mbps), High Performance LAN (~1 Gbps).

f. Network Data Base Management System

Therefore, any computer system that meets these protocol requirements, as well as the DoD ruggedization requirements, will qualify as an NGCR candidate. In preparation for NGCR availability, NAVSEA wishes to gain some experience in the development of Combat System software on commercially available machines that meet the above standards.

Task Description

Implement the basic features of a Combat Direction System on a commercially available microprocessor workstation.

Specific Requirements

The Navy desires to develop a low cost Combat Direction System (CDS) that can potentially be installed on non combatant ships or to augment the processing capability on board current CDS - equipped ships. An incremental approach to the overall system development will help insure that this effort provides maximum benefit to the Navy in terms of experience and potential end product use. A total of five increments, with a demonstration at the end of each increment, will result in a continuously improving product that reflects the desired features of the CDS community. The specific features of the five increments are detailed in the following paragraphs.

Increment One - Select the computer system, run-time operating system, and software support environment for the LCCDS. Design/develop or select an existing object oriented data base manager that interacts with the entire CDS data base (as it gets defined) and allows the user to define new objects that are not part of the CDS data base. The data base manager should provide features for data entry, user friendly query language for building logical and arithmetic relationships between data base elements, and a powerful output generator for developing display screen and hard copy formats. Design/develop a display/graphics capability which interacts with the data base manager and provides the user with the building blocks necessary to define his own customized screen formats for interactive operation of the system. Display features should include but not be limited to the following:

a. Windows and pull down menus for operation of all program features including the operational features such as mode selection, range scale, track information, help commands, display doctrine activation and deactivation.

b. Window and pull ɯown menu selection controlled via mouse, trackball, light pen, touch screen or any other feature deemed useable by the developer.

c. Graphics capability to display all symbology defined in ɘnclosure 4 of this package.

d. Ability to overlay the track and ownship position portion of the display with world vector shoreline maps as available from the Defense Mapping Agency (DMA).

All pull down menu options and window spaces should be directly coupled with the data base manager such that all information can be user defined and changed as required. That is, the display capability should be built in a generic fashion such that the user can tailor all display presentations including all pull down menu options and window spaces effectively building a new run time Man Machine Interface (MMI) as desired.

Display doctrine is a feature which enables the user to define the conditions under which data will be displayed and how to present the displayed data. Doctrine statements should be IF - THEN - ELSE type statements where the IF clause provides for operations on tactical information such as type, speed and bearing of tracks to be displayed. The THEN clause provides for the data presentation such as blink the symbol, enlarge, bold type, etc. Doctrine should be simple to define and easily activated or deactivated. Doctrine statements should be allowed to be defined individually and combined, if desired, into a doctrine tree which consists of up to about twelve separate statements. The IF clause of a doctrine statement should allow for at least twelve qualifiers.

The general response time to any user selected display option should be no greater that one half second.

Increment Two - Integrate a Manual Tracking and Identification (ID) capability to the basic display capability. Manual Tracking and ID refers to the ability of the system user to build and display a set of geographically stable and/or moving points of information of the position portion of the display. Manual Tracking and ID will include but not be limited to the following features:

a. Maintain the ownship symbol in the center of the position display at all times.

b. Introduce a standard display symbol (see enclosure 4 of the package) at the current location of the cursor. All symbols should be maintained relative to the ownship symbol.

c. Allow user to assign a speed and bearing to a vehicular track. Display a proportioned speed leader on the track and change its position at least once every 4 seconds. Track position should be dead reckoned using the current track bearing.

d. Allow user to assign additional information to any type of track.

e. Allow a currently displayed track to be hooked (selected) and display selected additional information in a window (e.g. for a hooked air track display speed, bearing, type, route, etc.). This additional information will be an object from a user defined window.

f. Allow user to change the ID of a track. (e.g. from unknown to friendly).

g. Provide for an expandable track file with no artificial size limitations.

Increment Three - Integrate a receive only Link 11 capability which provides for the receipt and display of track information from the Ship to Ship digital data link. This increment will entail the development of an interface to a Link 11 system via NTDS protocol, parsing link messages and displaying parsed track information on the position display. It is not expected that the LCCDS will package and ship locally generated tracks for output on the data link. This will be a receive - only system. The specific format of the Link 11 data is specified in a classified Interface Design Specification (IDS) and Operations Specification and will be provided under a different cover. For quote purposes assume twelve message types and six variations on each message type. NTDS interface boards are commercially available and will not require a hardware development effort.

Increment Four - Integrate an ownship maneuvering capability which includes navigation capability and track interface information. Ownship maneuvering will include but not be limited to the following:

a. Allow for the specification of up to six steaming routes with up to 50 waypoints (destinations) per route.

b. Provide Closest Point of Approach (CPA) geometry from ownship to any track and between any two tracks. Display bearing lines on the position display.

Increment Five - Integrate an organic auto tracking capability using the (TBD) radar interface. This task entails the development of an interface to a ship mounted radar, parsing the input information and displaying the radar contact information of the position display. This feature should not be costed at this time but the implementation should be considered for easy add on at a later date. A radar IDS will be provided under a different cover.

APPENDIX B

TABLE OF REFINED GOALS

G 1: The Man-Machine Interface 36

G 1.1: MMI Building Blocks 36

G 1.1.1: Tactical Display Architecture 36

G 1.1.1.1: Window Classes 38

G 1.1.1.1.1: Functions I/O 38

G 1.1.1.1.2: Graphics I/O 39

G 1.1.1.1.3: Alphanumeric I/O 39

G 1.1.1.2: Basic Conventions for the MMI 39

G 1.1.1.2.1: User Interaction Techniques 39

G 1.1.1.2.1.1: Trackball Interaction Terminology 40

G 1.1.1.2.1.2: Basic Interactive Window Features 40

G 1.1.1.2.1.2.1: Basic Window Interaction 41

G 1.1.1.2.1.2.2.: Window Manipulation 44

G 1.1.1.2.1.2.3: Window States 45

G 1.1.1.2.1.2.4: Window Formats 46

G 1.1.1.2.1.2.5.: Menus 48

G 1.1.1.2.1.2.6: Window Layout Requirement 49

G 1.1.1.2.1.2.7 Keyboard Guidelines 49

G 1.1.1.2.1.2.8: Help Guidelines 49

G 1.1.1.2.1.2.9: General Functionality Guidelines 49

G 1.1.1.3: LCCDS Windows 49

G 1.1.1.3.1: The Root Window 49

G 1.1.1.3.2: The Tactical Plot Window 50

G 1.1.1.3.3: The Radar and Video Select Window 50

G 1.1.1.3.4: The Interface Control Window 51

G 1.1.1.3.5: New Track Establishment Window 51

G 1.2: Tactical Display Features 53

G 1.2.1: Tactical Plot (TacPlot) 54

G 1.2.1.1: NTDS Standard Symbology 54

G 1.2.1.2: Track Category Select Function 59

G 1.2.1.3: TacPlot Display Control Functions 59

G 1.2.1.4: TacPlot Graphic Tools 62

G 1.2.1.5: Map Display 65

G 1.2.1.5.1: Map Database 65

G 1.2.1.5.2: Map Functions 65

G 1.2.1.5.2.1: Map Display Function 65

G 1.2.1.5.2.2: Map Select Function 65

G 1.2.1.5.2.3: Bottom Contours 65

G 1.2.1.5.2.4: Display Grid Lines 66

G 1.2.1.5.2.4.1: Coordinate Labels 66

G 1.2.1.5.2.5: Landmass Fill 66

G 1.2.1.5.2.6: Refresh 66

G 1.2.1.5.2.6.1: Map Refresh 66

G 1.2.1.5.2.7: Map Clear 66

G 1.2.1.6: Sensor Selection 66

G 1.3: Display Doctrine 67

G 1.3.1: Display Doctrine Scope 67

G 1.3.2: Display Doctrine Structure 67

G 1.3.2.1: The IF Clause 68
G 1.3.2.1.1: Logical Connectives 68
G 1.3.2.1.2: IF Clause Qualifying Conditions 68
G 1.3.2.1.3: Boolean Expression List 69
G 1.3.2.1.4: Expression Attributes and Threshold Values 69
G 1.3.2.2: The THEN Clause 70
G 1.3.2.2.1: THEN Operations Definition 70
G 1.3.2.2.1.1: Default Operations 70
G 1.3.2.2.1.2: User-Defined Operations 71
G 1.3.3: Display Doctrine Modification 72
G 1.3.4: Display Doctrine Consistency Checking 72
G 1.3.5: Display Doctrine Applications 72
G 2: System Control Function 74
G 2.1: Initialization: 74
G 2.1.1: Ownship Position 74
G 2.1.2: Greenwich Mean Time 74
G 2.1.3: CPA Parameters 74
G 2.1.4: Data Link Reference Point (DLRP) 75
G 2.1.5: Julian Date 75
G 2.1.6: Initial Values 75
G 2.2: Configuration Control 75
G 2.2.1: I/O Channels 75
G 2.3: Configuration Monitoring 76
G 2.3.1: LCCDS System Status. 76
G 2.3.2: Software Monitoring 76
G 2.3.3: Recovery Data 76

G 2.4: Training Function 76

G 2.5: Test Function 77

G 3: Tactical Database 78

G 3.1: Integrate OODBMS into LCCDSWS 78

G 3.2: Provide a Set of Object Types 78

G 3.2.1: Types of Instantiated Objects 79

G 3.3: Definition of New Classes 85

G 3.4: Provide Input and Output for OODBMS 87

G 3.4.1: Interaction with MMI 87

G 3.4.1.1: MMI Input 87

G 3.4.1.1.1: Graphical Input 87

G 3.4.1.1.2: Alphanumerical Input 87

G 3.4.1.1.3: Functional Input 88

G 3.4.1.2: MMI Output 88

G 3.4.1.2.1: Graphical and Alphanumerical Output 88

G 3.4.2: Interaction with Tracking System 88

G 3.4.3: Interaction with Radar Interface 88

G 3.4.4: Interaction with Link 11 Interface 88

G 3.4.5: OODBMS System Alerts 89

G 3.5: Query Language 89

G 4: The Basic CDS Functions 89

G 4.1: Ownship Monitor Function 90

G 4.1.1: Monitor Ownship Position 90

G 4.1.1.1: Manual Inputs 90

G 4.1.1.2: Automatic Inputs 90

G 4.1.1.3: Navigational Computation 90

G 4.1.2: Monitor Ownship Surface Maneuvering 91
G 4.1.2.1: Waypoint Geometry 91
G 4.1.2.2: Closest Point of Approach Processing 91
G 4.1.3: Monitor Surface Area Management 92
G 4.1.3.1: Special Points 92
G 4.1.3.1.1: Man in Water 92
G 4.1.3.1.2: Formation Center and PIM 92
G 4.1.3.1.3: Navigation Hazards and Reference Points 93
G 4.1.3.1.4: DLRP 93
G 4.1.3.2: Chart Processing 93
G 4.2: Tracking Function. 93
G 4.2.1: New Track Establishment 93
G 4.2.2: Track Position Data 94
G 4.2.2.1: Track Position Update 94
G 4.2.2.2: Track Reposition 94
G 4.2.3: Track Course and Speed Determination 94
G 4.2.3.1: Course and Speed Computations 94
G 4.2.3.2: Manual Course and Speed Entries 94
G 4.2.4: Dead Reckoning 95
G 4.2.5: Track Position Prediction: 95
G 4.2.6: Track History Processing 96
G 4.2.7: Track Termination 95
G 4.2.7.1: Automatic Termination 96
G 4.2.7.2: Manual Termination 96
G 4.3: Identification Function 96
G 4.3.1: Track Category Determination 96

G 4.3.1.1: Initial Category Assignment 96
G 4.3.1.2: Category Change Processing 97
G 4.3.2: Track Identity Determination 97
G 4.3.2.1: Initial Vehicular Track Identity Assignment 97
G 4.3.2.2: Vehicular Track Identity Change Processing 97
G 4.4: Search and Detection Function 97
G 4.4.1.: Radar Interface 97
G 4.4.2.: Interface Radar and OODBMS 97
G 4.5: Communication Function. 98
G 4.5.1: Link 11 98
G 4.6: Tableau System 98
G 4.6.1: Basic Features 98
G 4.6.2: Tableau Representation 98
G 4.6.3: Tableau Types 99
G 4.6.3.1: Index 99
G 4.6.3.2: Navigation 100
G 4.6.3.3: PIM 101
G 4.6.3.4: Float Plan 101
G 4.6.3.5: Communications 101
G 4.6.3.6: Active Tracks 102
G 4.6.3.7: Selected Track 102
G 4.6.3.8: Display Doctrine 102
G 4.6.3.9: Data Link Control 102
G 4.6.3.10: System Status 102
G 4.6.3.11: User Defined Tableaux 103
G 5: Implementation in Steps 103

G 6: LCCDSWS is highly Concurrent 103

G 6.1: Provide Real-Time Scheduling and Allocation 104

G 6.2: Process Load Orders in Real-Time 104

LIST OF REFERENCES

1. Swenson, E. N., Mahinske, E. B., Stoutenburgh, J. S., "NTDS - A Page in Naval History", Naval Engineers Journal, pp. 53-61, May 1988.

2. Department of the Navy, (NAVSEA) 0967-LP-027-8602, Systems Engineering Handbook Vol. I, Combat Direction System Model 5, February 1985.

3. Naval Research Advisory Committee, "Next Generation Computer Resources", Report presented to CNO, SPAWAR, NAVAIR, NAVSEA, February 1989.

4. Commander, Naval Sea Systems Command UNCLASSIFIED Letter 9410 OPR:61Y Serial 61Y/1036 to Superintendent, Naval Postgraduate School, Subject: Statement of Work for Low Cost Combat Direction System, 20 December 1988.

5 Space and Naval Warfare Systems Command, (UNCLASSIFIED) Software Requirements Specification for the NCCS-A Workstation Executive, v. 1, Man-Machine Interface, 23 June 1989.

6. V. Berzins, "Software Engineering", class notes provided at US Naval Postgraduate School, Monterey, CA, Summer Quarter 1989.

7. Department of the Navy, Military Specification (CONFIDENTIAL NAVSEA) 0967-LP-027-8635, Combat Direction Systems (CDS) Specification for Surface Ships (Model 4.1) (U), Vol. 1, Revision 5, April 1988.

8. Interview between W. Hanley, Deputy Director, Surface Combat Directions Systems (Code 2), Fleet Combat Direction Systems Support Activity, San Diego, CA, and the authors, 19 July 1989.

9. Department of the Navy, Military Specification, (CONFIDENTIAL NAVSEA), 0967-LP-027-8635, Combat Direction Systems (CDS) Specification for Surface Ships (Model 4.1) (U), Vol. 2, Revision 5, April 1988.

10. Enns, Jamie G., "Commercial Workstations Offer Opportunities and Challenges in Tactical Arena", *Defense Computing*, p. 22, July-August 1989.

11. Sun Microsystems, Sun System Overview, February 1986.

12. Lapin, J. E., *Portable C and UNIX System Programming*, Prentice Hall, Inc., 1987.

13. Fleet Combat Direction System Support Agency, (CONFIDENTIAL) Joint Fleet Combat Direction Systems Standardization Manual 74-011(U), Section 6, Revision 5, April 1975.

14. Jones, Oliver, *Introduction to the X Window System*, Prentice Hall, Inc., 1989.

15. Naval Air Development Center, Technical Manual (UNCLASSIFIED) NAVAIR 01-75PAC-11-3-1, Software Reference Manual for Tactical Coordinator, Navy Model P-3C Update Aircraft, 1 August 1986.

16. Ross, Debrah, "An Object Oriented Data Base Manager for the Low Cost Combat Direction System", Masters Thesis, US Naval Postgraduate School, Monterey Ca., December 1989.

17. Booch, Grady, *Software Engineering with Ada*, The Benjamin/Cummings Publishing Company, 1986.

18. Yourdon, Edward, *Modern Structured Analysis*, Yourdon Press, 1989.

19. Baecker, R. M., Buxton, W. S., *Readings in Human-Computer Interaction: A Multidisciplinary Approach*, Morgan Kaufmann Publishers, Inc., 1987.

20. Card, S. K., Moran, T. P., Newell, A., *The Psychology of Human-Computer Interaction*, Lawrence Erlbaum Associates, Inc., 1983.

21. Baecker, R. M., Buxton, W. S., *Readings in Human-Computer Interaction: A Multidisciplinary Approach*, "Color Graphics - Blessing or Ballyhoo?", p. 333, Morgan Kaufmann Publishers, Inc., 1987

22. Deatherage, B. H. , "Auditory and Other Sensory Forms of Information Presentation, in Van Cott, H.P. & Kinkade, R. G. (eds), *Human Engineering Guide to Equipment Design, Revised Edition*, U. S. Government Printing Office, 1972.

23. Poock, Gary K., "Speech Recognition Research, Applications and International Efforts", Proceedings of the Human Factors Society 30th Annual Meeting, pp.1278-1283, October 1986, Dayton, Ohio.

24. Norcio, A. F. and Stanley, J., "Adaptive Human-Computer Interfaces", Naval Research Laboratory (NRL) Report 9148, 30 September 1988.

25. Greenberg, S and Witten, L. H., "Adaptive Personalized Interfaces - A Question of Viability", Behavior and Information Technology 4(1), pp. 31-45, 1985.

26. Luqi, and Berzins, V., "Rapidly Prototyping Real-Time Systems," IEEE Software, v. 5, pp. 25-36, September 1988.

27. LuQi, and Ketabchi, M. " A Computer Aided Protoyping System" IEEE Software, March 1988, pp. 67-72.

28. Moffit, C.R. II, "A Language Translator For A Computer Aided Rapid Prototyping System", Master's Thesis, Naval Postgraduate School, Monterey, Ca, September 1988

29. Interview between CDR Rich Gragg, Chief Of Naval Operations (OP-942F4), Washington, DC, and the authors, 1 May 1990.

30. Interview between Dr. Glenn Osga, Naval Ocean Systems Center (Code 44), San Diego, CA, and the authors, 29 March 1990.

31. Unisys, AN/UYK-43(V) Technical Description, March 1987.

INITIAL DISTRIBUTION LIST

Defense Technical Information Center Cameron Station Alexandria, VA 22304-6145	2
Dudley Knox Library Code 0142 Naval Postgraduate School Monterey, CA 93943	2
Director of Research Administration Code 012 Naval Postgraduate School Monterey, CA 93943	1
Chairman, Code 52 Computer Science Department Naval Postgraduate School Monterey, CA 93943	1
Office of Naval Research 800 N. Quincy Street Arlington, VA 22217-5000	1
Center for Naval Analysis 4401 Ford Avenue Alexandria, VA 22302-0268	1
National Science Foundation ATTN: Dr. K. C. Tai Division of Computer and Computation Research Washington, D.C. 20550	1
Office of the Chief of Naval Operations Code OP-941 Washington, D.C. 20350-2000	1
Office of the Chief of Naval Operations Dr. John R. Davis, Chief Scientist Code OP-094H Washington, D.C. 20350-2000	1

Office of the Chief of Naval Operations 1
Code OP-945
Washington, D.C. 20350-2000

Office of the Chief of Naval Operations 1
ATTN: CDR R. V. Gragg
Code OP-942F4
Washington, D.C. 20350-2000

Commander 2
Naval Sea Systems Command
ATTN: LCDR Scott Kelly
Code 06D3131
Washington, D.C. 20362-5101

Commander 2
Naval Telecommunications Command
Naval Telecommunications Command Headquarters
4401 Massachusetts Avenue NW
Washington, D.C. 20390-5290

Commander 1
Naval Data Automation Command
Washington Navy Yard
Washington, D.C. 20374-1662

Commander 2
Space and Naval Warfare Systems Command
ATTN: CAPT John Gauss
PMW-162
Washington, D.C. 20363-5100

Assistant Secretary of the Navy 1
Research, Development and Acquisitions
Washington, D.C. 20350-1000

Dr. Lui Sha 1
Carnegie Mellon University
Software Engineering Institute
Department of Computer Science
Pittsburgh, PA 15260

Commanding Officer 1
Code 5150
Naval Research Laboratory
Washington, D.C. 20375-5000

Defense Advanced Research Projects Agency (DARPA) 1
Director, Naval Technology Office
1400 Wilson Boulevard
Arlington, VA 2209-2308

Defense Advanced Research Projects Agency (DARPA) 1
Director, Prototype Projects Office
1400 Wilson Boulevard
Arlington, VA 2209-2308

Defense Advanced Research Projects Agency (DARPA) 1
Director, Tactical Technology Office
1400 Wilson Boulevard
Arlington, VA 2209-2308

Dr. R. M. Carroll (OP-01B2) 1
Chief of Naval Operations1
Washington, DC 20350

Dr. Robert M. Balzer 1
USC-Information Sciences Institute
4676 Admiralty Way
Suite 1001
Marina del Ray, CA 90292-6695

Dr. Ted Lewis 1
Editor-in-Chief, IEEE Software
Oregon State University
Computer Science Department
Corvallis, OR 97331

Dr. R. T. Yeh
International Software Systems Inc. 1
12710 Research Boulevard, Suite 301
Austin, TX 78759

Dr. C. Green 1
Kestrel Institute
1801 Page Mill Road
Palo Alto, CA 94304

Prof. D. Berry 1
Department of Computer Science
University of California
Los Angelas, CA 90024

Director, Naval Telecommunications System Integration Center 1
NAVCOMMUNIT Washington
Washington, D.C. 20363-5110

Dr. Knudsen 1
Code PD50
Space and Naval Warfare Systems Command
Washington, D.C. 20363-5110

Ada Joint Program Office 1
OUSDRE(R&AT)
The Pentagon
Washington, D.C. 23030

CAPT A. Thompson 1
Naval Sea Systems Command
National Center #2, Suite 7N06
Washington, D.C. 22202

Dr. Peter Ng 1
New Jersey Institute of Technology
Computer Science Department
Newark, NJ 07102

Dr. Van Tilborg 1
Office of Naval Research
Computer Science Division, Code 1133
800 N. Quincy Street
Arlington, VA 22217-5000

Dr. R. Wachter 1
Office of Naval Research
Computer Science Division, Code 1133
800 N. Quincy Street
Arlington, VA 22217-5000

Dr. J. Smith, Code 1211 1
Office of Naval Research1
Applied Mathematics and Computer Science
800 N. Quincy Street
Arlington, VA 22217-5000

Mr. William E. Rzepka 1
U.S. Air Force Systems Command
Rome Air Development Center
RADC/COE
Griffis Air Force Base, NY 13441-5700

Dr. C.V. Ramamoorthy 1
University of California at Berkeley
Department of Electrical Engineering and Computer Science
Computer Science Division
Berkeley, CA 90024

Dr. Nancy Levenson 1
University of California at Irvine
Department of Computer and Information Science
Irvine, CA 92717

Mr. George Roberson 1
Fleet Combat Directional Systems Support Activity
San Diego, CA 92147-5081

Mr. William Hanley 1
Fleet Combat Directional Systems Support Activity
San Diego, CA 92147-5081

Dr. Earl Chavis (OP-162) 1
Chief of Naval Operations
Washington, DC 20350

Dr. Alan Hevner 1
University of Maryland
College of Business Management
Tydings Hall, Room 0137
College Park, MD 20742

Dr. Al Mok 1
University of Texas at Austin
Computer Science Department
Austin, TX 78712

George Sumiall 1
US Army Headquarters
CECOM
AMSEL-RD-SE-AST-SE
Fort Monmouth, NJ 07703-5000

Mr. Joel Trimble 1
1211 South Fern Street, C107
Arlington, VA 22202

Linwood Sutton 1
Code 423
Naval Ocean Systems Center
San Diego, CA 92152-5000

Dr. Sherman Gee 1
Code 221
Office of Naval Technology
200 N. Quincy St.
Arlington, VA 22217

Dr. Mark Kellner 1
Carnegie-Mellon University
Software Engineering Institute
Pittsburgh, PA 15213

Luqi 50
Code 52Lq
Computer Science Department
Naval Postgraduate School
Monterey, CA 93943

Dr. Valdis Berzins 1
Code 52Bz
Computer Science Department
Naval Postgraduate School
Monterey, CA 93943

Commanding Officer 1
Naval Command and Control Systems Command
Wibbelhof St. 3
2340 Wilhelmshaven
West Germany

Commander 1
Naval Ocean Systems Center
Code 43
San Diego, CA 92152-5000
ATTN: CDR Wayne Duke

Commander 1
Naval Ocean Systems Center
Code 44
San Diego, CA 92152-5000
ATTN: Dr. Glenn Osga

Dr. Chuck Hutchins 1
Code 55Hu
Operations Research Department
Naval Postgraduate School
Monterey, CA 93943

Dr. Gary Poock 1
Code 55Pk
Operations Research Department
Naval Postgraduate School
Monterey, CA 93943

LCDR James A. Seveney 2
8507 Shirley Woods Ct.
Lorton, VA 22079

Commanding Officer 2
1st Destroyer Squadron
Mecklenburger St. 50 A I
2300 Kiel
West Germany
ATTN: LCDR Guenter P. Steinberg